Planning for Army Force Generation Using Lot Streaming, and Extensions

Adria Elizabeth Markowski

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Industrial and Systems Engineering

Subhash C. Sarin, Chair,
Douglas R. Bish
John B. Halstead
C. Patrick Koelling

November 4, 2011
Blacksburg, Virginia

Keywords: lot streaming, flow time, wait time, optimization, column generation, Dantzig-Wolfe, batching, lot splitting, hybrid flow shop, agile manufacturing, ARFORGEN

Planning for Army Force Generation Using Lot Streaming, and Extensions

(ABSTRACT)

Adria Elizabeth Markowski

As the Army transitions to the Army Force Generation (ARFORGEN) deployment cycle, it must adjust its many operations in support of ARFORGEN. Specifically, the Initial Military Training (IMT) must be able to adjust the scheduling of its classes for newly enlisted service members to finish training such that they fulfill Brigade Combat Team (BCT) requirements within their common due windows. We formulate this problem as a lot streaming problem. Lot streaming splits a batch of jobs into sublots,which are then processed over the machines in an overlapping fashion. To schedule classes for the IMT, there are two stages that must be coordinated: Basic Training (BT) and Advanced Individual Training (AIT). For the Army Force Generation problem, the classes are considered as sublots that are streamed from one stage to the next. For this process, the model formulation must address determination of class sizes and scheduling of soldiers and classes at the two stages such that (1) the start times of the soldiers at Stage 2 are greater than their completion times at Stage 1, and (2) the assignment of requisite number of soldiers is made to each BCT, so as to minimize the total flow time.

We propose a decomposition-based approach for the solution of this problem. In an effort to decompose the problem, the original lot streaming problem is reformulated such that the master problem selects an optimal combination of schedules for training classes and assigning soldiers to BCTs. A complete schedule selected in the master problem includes the assignments of soldiers to classes in BT, AIT, and their assignments to the BCTs, so as to minimize the total flow time as well as earliness and tardiness for regular Army units. Earliness and Tardiness are defined as the length of the time a soldier waits before and after the due date, respectively, of the BCT to which he or she is assigned. Our decomposition-based method enables solution of larger problem instances without running out of memory, and it affords CPU time reductions when compared with the CPU times required for these problem instances obtained via direct application of CPLEX 12.1.

Our investigation into the structure of the problem has enabled further improvement of the proposed decomposition-based method. This improvement is achieved because of a result, which we show, that the first and second-stage scheduling problems need not be solved as one combined subproblem, but rather, they can be solved sequentially, the first stage problem followed by the second stage problem. The combination of Stage 1 and Stage 2 problems as one subproblem creates several additional enumerations of possible schedules the model must generate. By reducing this number of enumerations, the computational effort involved in solving the model reduces significantly, thereby allowing reductions in CPU time. In the Sequential approach, the completion times of soldiers determined at Stage 1 are passed to Stage 2 as bounds on their completion times at Stage 2. We prove that solving the combined subproblem sequentially as two subproblems is optimal when the first stage has no limit on the batch size and the ready times of the soldiers at Stage 1 are the

same. For the Army Force Generation problem, we use unequal ready times, and therefore, solving the scheduling problems for the first two stages as sequential subproblems can lead to suboptimal solutions. Our experimental investigation shows efficacy of solving larger-sized problem instances with this method. We also recommend various potential additions to improve the Sequential approach for application to the overall Army problem. We have also demonstrated the use of our methodology to a real-life problem instance. Our methodology results in schedules for IMT with an estimated 28% reduction in mean flow time for soldiers over what is currently experienced in practice.

We apply this Sequential approach to various extensions of the problem on hand that pertain to hybrid flow shop and agile manufacturing environments. Results of our computational investigation show the effectiveness of using the Sequential approach over direct solution by CPLEX from the viewpoint of both optimality gap and the CPU time required. In particular, we consider two different model configurations for a hybrid flow shop and three different model configurations for an agile manufacturing facility.

Dedication

For my family

Acknowledgments

"I can do all things through Christ who strengthens me." From the start, this journey has been a series of impossible events, that I must now redefine what I think to be impossible. For this I must first thank God for my abilities, opportunities, and strength to face every mountain that was on this journey.

A heart felt thanks to my advisor, Dr. Subhash Sarin, one of the most patient people I have had the pleasure of knowing. His work ethic and commitment to students are truly inspiring. His enduring calm demeanor has helped refocus my energies from constantly worrying about what I cannot do tomorrow but what is the best I can do today. His meticulous attention to detail and articulate ability to best present information has truly improved this dissertation. After three classes and countless hours of research meetings, it is impossible to quantify how much I have learned from Dr. Sarin.

An equally warm thanks to Dr. John Halstead. He has offered his support from the very beginning of this adventure. His support has been a tremendous asset in acquiring Army data across commands and senior leaders. In addition, he could always be counted on for encouraging words throughout my trials and tribulations. I would also like to thank my other two committee members, Dr. Patrick Koelling and Dr. Douglas Bish for their commitment, support and cooperation to help me meet my employer completion time. Their willingness to accomplish milestones during the summer helped make it possible to meet deadlines.

The opportunity to attend school was possible because of the amazing university training program provided by CP16. Thanks to this program I was able to be completely dedicated to studies while at school. Of course without the support from my employer, Army Training Support Center allowing me to attend university training, I would never have enjoyed the benefits of the program CP16 sponsors. I would especially like to thank my supervisor, George Baker for his flexibility and understanding of all the logistical challenges that occurred during the process. A warm thanks to all my co-workers on the Analysis team as well for carrying the load of work during my absence. Each of them, Bill Andersen, Doug Prior, Gene Frament, Ken Sampson, Randy Holmes, and Sam Epstein are a wealth of knowledge and no question was ever too small or too big. A special thank you to the analysts at TRADOC G-3/5/7 (Tony White and LTC Bill Emge) for their support and data collection. A special thanks to all my point of contacts at Army Human Resource Command, Army Accessions Command, Army G-1, and employees of Booz Allen Hamilton for sharing their knowledge and data.

A very heartfelt thank you to my best friend, Nikki Lewis who made living arrangements in Blacksburg a joy while I lived away from my husband and family. Her encouragement and incredible ability to bake kept me positive and ready to take on all the challenges that come with graduate school. A thanks to her husband Steve Huff who could always be counted on for a much needed laugh! I would also like to thank my close friend Allison Osborne who never let me wallow (too long) in the impossibility of any situation. Her continual encouragement and good listening skills were unfailing. Even though we were miles apart she always managed to include me in events at home to help keep homesickness at bay.

I would like to thank all my new friends at Virginia Tech. I would especially like to thank Jason and Laura Judd (my first new friends in Blacksburg). They could always be counted on for pleasurable company, great food and even better desserts. I thank Jason for all the homework team work, study sessions and research discussions. He spent tons of time showing me how to script in OPL that saved me time and headaches. I would also like to thank Esra Ağca for all the homework sessions and projects we worked on together. Classes were much more fun with her around! A special thanks to my office mate Anna Schuh for hearing my every complaint and always there to help, cheerup or force me to take breaks and getting me to see some Hokie football. Hours and hours of work at the office always passed by quickly and with several more laughs thanks to her. I would also like to thank all my other VT friends and classmates, Ed Chamberlayne, Andrew Henry, Victor Pereira, Fadel Megahead, and Kammy Mann. Thanks for getting me out of the office and enjoying some of Blacksburg! Thanks to Kelli Karcher for helping me get through real analysis and spending countless number of hours at the Math Empo. Thanks also to my professor at Christopher Newport University, Dr. Brian Bradie, for helping to prepare me for real analysis. I would like to thank my former supervisor, Dr. Joe Knickmeyer. No matter the question he was always able to help illuminate the cloudy areas of several different subjects.

Thanks to all the members of my church, Lighthouse Community Church. Their support and prayers carried me to the finish line and I love them all for it. Thanks also to my church at school, Prices Fork Methodist Church, especially Joyce and Ronnie and all the times they treated me to lunch. Lastly, but certainly not least, I want to thank my family. From my Great-Aunt Yetta's constant prayers, the support of my grandparents to family dinners with my brother and sister-in-law when I was back in town, to all the gifts from my other brother and sister-in-law in Japan, to care packages sent from my mother-in-law (nothing beats her apple pie filling) and all the cards and treats my mom sent to me while at school; their support gave me strength to continually face each day. Ultimately, there are no words in any language that could possibly express my gratitude to the unwavering love and support from my husband, David Markowski. From the first conversation about going on for school, he has constantly been my rock that I lean on constantly. I know I am stronger and more confident because of the faith he has in me. There is no boundary to what he does for me, whether it's building a computer that meets the specifications of all my modeling requirements, running updates on my computer to make sure it's always working, creating a system so that my dissertation and work are backed up on a daily basis no matter how far I am from home, or taking care of the house and animals by himself during the semesters I was away at school. He's one of the most giving people I know and I am truly blessed to be his wife.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Several Army Commands work in concert to obtain, train and distribute soldiers as resources to meet the Army's requirements. Individuals progress through different programs that constitute different paths. Officers assess through the United States Military Academy (USMA, [1]), Reserve Officer Training Corps (ROTC) or Officer Candidate School (OCS). The enlisted individuals are recruited by the US Army Accessions Command and sign a contract for a length of service and Military Occupational Specialty (MOS). The U.S. Training and Doctrine Command (TRADOC) manages the training aspect of the soldiers' careers. The U.S. Human Resource Command manages and distributes soldiers based on the resource demands and it is one of its responsibilities as "Army's functional proponent for the military personnel management system" [1].

[1] Please see Appendix A for a full listing of military acronyms

1.1.1 U.S. Army Accessions Command

USAAC's objective "is to provide the quantity and quality of volunteers to meet Army requirements" [1]. USAAC's responsibilities include the command and control of the recruiting process implemented to obtain the right quantity and quality of volunteers. The USAAC oversees the Recruiting Command and the Cadet Command. As part of the recruiting process, the recruiters qualify individuals for MOS assignments and create contracts for the enlisted future soldiers' terms of service.

1.1.2 U.S. Army Training and Doctrine Command

The U.S. Army Training and Doctrine Command (TRADOC) oversees all training related requirements from the creation to execution of training for the Army's soldiers and civilians consistent with the Command's mission. Available on the TRADOC website, its mission is "TRADOC develops the Army's soldier and civilian leaders and designs, develops and integrates capabilities, concepts and doctrine in order to build a campaign-capable expeditionary Army in support of joint war fighting commanders through Army Force Generation (ARFORGEN)." As indicated by the mission statement, the responsibilities of TRADOC include more than training-related duties. The desired capabilities that the Army leaders foresee have to be defined and integrated into training. TRADOC shapes the Army of the future by incorporating the desired or expected capability the Army should have into training and doctrine. Detailed by the Commanding General's vision of TRADOC, the command:

- recruits and trains soldiers,
- develops adaptive leaders,
- designs today's army modular force, and the future combat force and
- maximizes institutional learning and adaptation.

Consistent with TRADOC's mission of training soldiers and civilians, it operates 33 schools over 16 Army installations. Indicated by TRADOC's website, from 1 October 2007 through 30 September 2008, its schools ran 2,734 courses for a total of 503,164 seats for soldiers, civilians, and other-service personnel [67].

Figure 1.1: "Victory Starts Here" is TRADOC's motto. The classes that soldiers train in are not always traditional classroom settings [67].

Training Flow

The system diagram in Figure 1.2 illustrates new recruits training flow for enlisted soldiers and officers. Starting with the officer path, officers receive their commission and initial training by college graduation. Therefore, when officers begin their training in the active Army, they continue their basic training while also embarking on their specialty training. In the system diagram shown, the officers start their training courses depending on the requirements of their occupational specialty. Shown in the other path in Figure 1.1, after the enlisted volunteers join the Army, they report to Basic Combat Training (BT) to start their transformation from volunteers to soldiers, also called the Soldierization Process. During BT, new recruits remain together as the Soldierization training begins. After completing the nine weeks of BT, soldiers begin Advanced Individual Training (AIT) where soldiers begin learning for their Military Occupational Specialty (MOS). Each MOS has its own training plan that soldiers must complete before being qualified in their field and ready to be assigned

to a unit. Therefore, the length of time a soldier's training may take is dependent on the MOS, and also, on the Additional Skill Identifier (ASI) the soldier may be contracted to complete before unit assignment.

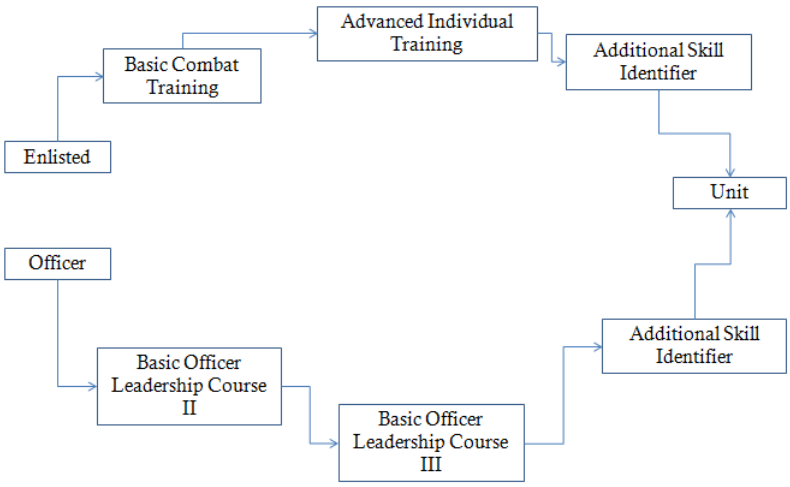


Figure 1.2: New Recruit Training Flow

Basic Combat Training of Enlisted Soldiers

Basic Combat Training spans over a period of nine weeks. During this time, enlisted soldiers move through three phases of training focused on transforming volunteers into soldiers. The three phases cover the basic instructions soldiers must master before moving on to train in their MOS in AIT. Among others, BT includes physical training as well as training in Army Culture, Warrior Ethos, history, teamwork, and Army Values [24]. For some MOSs, Basic Combat Training and AIT occur at the same installation. This is called One Station Unit Training (OSUT). Figure 1.3 shows the breakdown of training over the different phases and the goals associated with each phase.

PHASE		I	II	III	IV	V	V+
		Basic Combat Training (BCT)			Advanced Individual Training (AIT)		
		One Station Unit Training (OSUT)					
GOALS	WEEKS	1 THRU 3	4 THRU 6	7 THRU 9	10 THRU 13	14 THRU 20	21 THRU COMPLETION
PASS PHASE TRAINING		X	X	X	X	X	
CONFORM TO STANDARDS		X	X	X	X	X	X
OPERATE AS A TEAM MEMBER		X	X	X	X	X	X
MASTER BASIC SKILLS		X	X	X	X	X	X
DEVELOP PHYSICAL FITNESS		X	X	X	X	X	X
DEMONSTRATE SELF DISCIPLINE			X	X	X	X	X
DEMONSTRATE THE CAPABILITY TO COPE WITH STRESS		X	X	X	X	X	X
IMMERSE IN ARMY VALUES		X	X	X			
DEMONSTRATE ARMY VALUES				X	X	X	X
QUALIFY WITH RIFLE			X				X
PASS APFT				X[1]	X	X	X
CONDUCT CLFX				X	X	X	X
CONDUCT URBAN OPERATIONS				X	X	X	X
CONDUCT COMBATIVES		X	X	X	X	X	X
CONDUCT WTBD		X	X	X	X	X	X
COMPLETE POI REQUIREMENTS				X	X	X	X

[1] BCT requirement for APFT is 50 points per event.

Figure 1.3: Initial Enlisted Training Schedule.

Advanced Individual Training

Advanced Individual Training (AIT) occurs after enlisted soldiers successfully complete BT, and it is shown in Figure 1.3 as Phases IV, V and V+ of Individual Enlisted Training (IET). During AIT, soldiers receive reinforcement of training received in BT and start training specific to their MOS. Different MOSs have different course requirements and different total training time, in order to be qualified.

Additional Skill Identifiers

Additional Skill Identifiers (ASI) are different training courses that soldiers have taken beyond the skills of their MOS. A soldier can complete the MOS requirements, and train for an ASI before joining a unit. Depending on the contract a solider signs upon entry, the Army may be required to ensure the soldier to receive training for the ASI before being assigned to a unit. Some unit positions may even require a soldier to take certain ASI before being assigned to a unit.

Officer Training

The Officer Education System (OES) includes the initial military training that all officers are required to take before being assigned to a unit. The Initial Entry Training for officers includes Pre-appointment and Pre-commissioning programs, and Basic Officer Leadership Course B (BOLC B) [25]. Similarly to the enlisted AIT, officers go through qualification training courses related to their Area of Concentration (AOC) during BOLC B.

1.1.3 Army Force Generation

ARFORGEN is a cyclic scheduling process the Army uses to maximize the number of units available to deploy in order to harness a surge capability, while not exceeding the supply of soldiers or a deployment tempo unsustainable by soldiers and their families ([41] and [22]). Before describing the ARFORGEN process in general, some background information is provided to fully appreciate the process. The force generating system the Army used since the beginning of the Cold War has undergone drastic transformation in the last decade. Under the Army's recent transformations, including the shift to modular forces and capabilities planning versus threat based planning, the Army developed the Army Force Generation Model. The model was created to increase readiness and mission capability through the use of a cyclic process [1]. Before the current Army Force Generation model was implemented, the Army used a divisional force structure that was comprised of several different types of specialized units [41]. The deployment and structure of divisions were based on the Table of Organization and Equipment (TOE), the Modified Table of Organization and Equipment (MTOE) and Operation Plans (OPLAN) [14]. Prior to the Global War on Terror (GWOT), the Army recognized the value of moving to a modular brigade-based force from the traditional divisional force in an attempt to achieve a better balance of fire power and deployability [6]. The divisional structure the Army used before the transition to modular units was large. It comprised of multiple units and was difficult to sustain over long periods

of time. The divisional force structure design satisfies the demands of more conventional warfare and narrower missions according to the Quadrennial Defense Review (QDR) Report from 2001 [27]. The divisional structure is also much less flexible than modular units and is not appropriate for long-term tasks. In the same QDR from 2001, the DOD recognized "the need to size U.S. military forces not only for the most demanding near-term warfighting tasks, but also, for a plausible set of other near-term contingencies" [27, page 60]. A pictorial comparison between the division-structured force and modular force is shown in Figure 1.4. The division-based organization has multiple pieces all with specific purposes moving as one group, completing one short-term task. The brigade-based organization has different brigade combat teams that can be assigned separate tasks or missions. The units that make up the division-based structure can not form into the mission specific units. This inflexible nature of the division-based structure is in stark contrast to the flexibility of the modular brigade-based structure. Hence, one of the major benefits of having a more modular-based organization is the flexibility of having different brigade combat teams performing separate missions within one division [42].

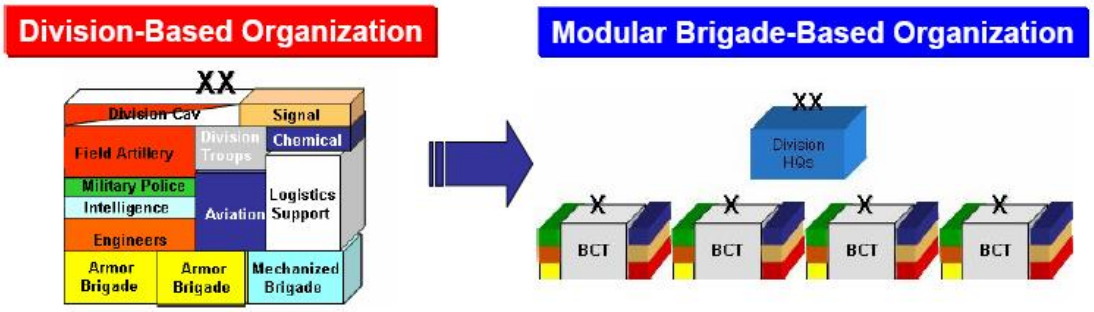


Figure 1.4: Division Compared to Modular Unit as shown in [42]

This transformation to modular brigade-based structures to meet warfighting tasks provides the potential for the Army to sustain continuous operations in the face of persistent conflict. The demands of this type of capability include a constant ready force as opposed to the former tiered readiness approach [41]. One of the results of ARFORGEN is the constant availability of units trained and ready to complete full spectrum operations [57]. In order

to have a constant ready force, the ARFORGEN process splits the total number of brigade combat teams into thirds such that, at any given time, one third of the brigade combat teams are available to be deployed as needed. ARFORGEN accomplishes this feat by following a three phase cycle. As shown in the ARFORGEN model shown in Figure 1.5, the brigade combat teams move through the following three phases.

- **Reset/Train Force**: Units returning from deployment enter the Reset/Train phase in order to man the unit and return old and receive new equipment. After returning from a deployment, many soldiers within a unit are assigned to a new unit for various reasons such as promotions. Some soldiers may stay within the unit; however, the soldiers that remain have training requirements and are given opportunities to take leave as they reunite with their families. During this time of individual training and leave, the unit must fulfill the personnel gaps created from departed soldiers before transitioning to the next phase [26].

- **Ready Force**: After meeting the capability requirements in the Reset/Train Force phase, units move to the Ready Force phase. In the Ready phase, units conduct collective training [26]. In collective training, soldiers of a unit work together and through team work they accomplish interdependent tasks. These tasks are performance-oriented, and are made to reflect the possible missions the unit may have when deployed. To move to the next phase, units must be fully trained in all the Mission-Essential Task List (METL) items [1].

- **Available**: After units are fully trained in their specified METL, they move to the Available phase. This phase is designed to have one-third of the total units (brigade combat teams) available to deploy when and wherever needed [26], the remaining being assigned to the Army manning requirements not following ARFORGEN.

ARFORGEN is event driven, guiding leaders to allocate resources to units based on an event. By being event driven, resource allocation is based on the requirements of a specific

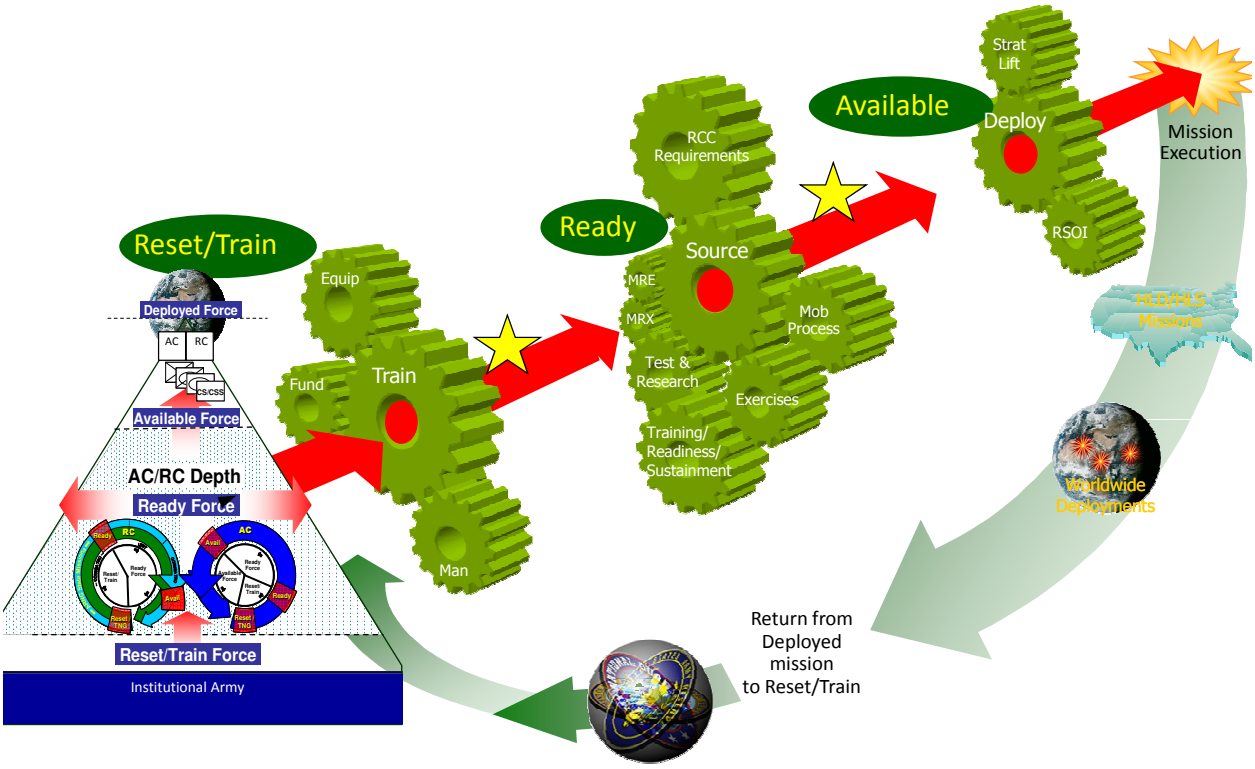


Figure 1.5: Army Force Generation Progression as shown in [57]

event (such as a mission), and is prioritized accordingly. Previously, if there were shortages, they would be distributed across units equally, decreasing the number of available forces. Without this decrease in available forces, the Army is more capable to meet the dynamic demands of the Combatant Commander more rapidly [26]. The cyclic progression makes the location and number of units in a particular phase more predictable, thus allowing for better allocation of the resources, such as the equipment or training that the units may need [57].

As the Army gradually moves to the ARFORGEN process, the supporting components that make the process possible must adjust from the former tiered readiness standard

to the cyclic nature of ARFORGEN. Each phase of ARFORGEN can be considered a system with components and challenges that comprise each system. During the first phase of ARFORGEN, the US Army Training and Doctrine Command (TRADOC) and US Army Human Resource Command (HRC) work in tandem to schedule soldiers for required individual training. HRC also assigns soldiers to new units as appropriate, and it manages the personnel requirements of the units.

One of the challenges that the two commands currently face is getting the right set of skilled soldiers to the unit in the time allotted for the Reset/Train phase. The unit must fulfill its personnel and equipment requirements based on the force structure specified in the MTOE. The MTOE lists the skill level, rank and number of soldiers required for each unit. HRC has to fulfill these requirements with available soldiers who are either in Reset with other units or are finishing institutional training. The soldiers finishing institutional training can be soldiers taking classes to advance their military careers. Other soldiers that can be finishing institutional training are new recruits and newly commissioned officers just completing Initial Military Training (IMT). Currently, the output of training schools that TRADOC manages, especially soldiers completing IMT, are too volatile for HRC to include in their assignment forecasts for units following ARFORGEN. A part of the volatility of soldiers completing IMT is caused by the volatile nature of how solders enter the system. Both enlisted and officer soldiers usually enter the training system soon after high school and college graduations, respectively. Therefore, a subcomponent of the Reset/Train phase worth mentioning is the US Army Accessions Command (USAAC). The USAAC is a subordinate command of TRADOC that manages the recruiting for the Army [1]. HRC gives USAAC recruiting goals based on forecasted skill gaps the Army is estimated to have. In an ideal world, the three organizations (components and subcomponent) would work seamlessly together to acquire, train, and assign the necessary soldiers to units during their Reset/Train phase while meeting the time constraints. In reality HRC and TRADOC are less than seamlessly coordinated

and often use ad hoc methods to get the necessary soldiers to the unit, sometimes not until the Ready phase of ARFORGEN.

During the second phase of ARFORGEN, the Ready phase, TRADOC is one of the components along with Forces Command (FORSCOM) and the Installation Management Command (IMCOM). Unlike the individual training that occurs during the Reset/Train phase, the collective training that takes place during the Ready phase trains the unit to work together as a team, accomplishing the mission through the skills of the unit. Since the tasks reflect those the unit may have while deployed, the training environment has to be as realistic as possible in order for the unit to achieve proficiency. Thus, TRADOC, FORSCOM and IMCOM have to work together to enable the appropriate training resources for the units to train. TRADOC and IMCOM provide the resources the units need to train such as instructors, ranges, training aids, devices, simulations, and simulators (TADSS). The units meet proficiency levels based on the commander's assessment. As owners of the units, FORSCOM monitors this process. Orchestrating the use of Battle Command Training Centers (BCTCs) and providing TADSS to units in enough time to meet proficiency before their scheduled Available phase start date has been challenging as the Generating Force adapts to ARFORGEN. TRADOC and IMCOM still must support the Army forces not yet following ARFORGEN while supporting the needs of the units. One purpose of ARFORGEN is to synchronize these requirements. Before ARFORGEN the Army systems that support and train the forces in collective training were not synchronized. The systems still are not synchronized and through paramount efforts of the components, there are ARFORGEN synchronization conferences that take place to make sure the units have the necessary training to move to the Available phase.

FORSCOM and HRC are the components during the Available phase of the ARFORGEN process. The unit deployment schedules following ARFORGEN are set by FORSCOM. Towards the end of the deployment, HRC starts scheduling training and determining the

units the soldiers will move to when the unit returns to the Reset/Train phase. One of the biggest challenges that faces the Available phase and the Ready phase is ensuring the units are fully manned. As the Army transitions to the ARFORGEN cycle, it is still adjusting to the timing of the manning requirements. In the tiered structure previously mentioned, if a unit was short, a resource was obtained from a unit with a later deployment date. With the Army's current missions, Army resources are under tremendous stress. Thus, at times, the shortages are still dealt with in the manner similar to the tiered structure. One solution to this challenge is ensuring enough personnel assigned to the unit as early as the Reset/Train phase and possibly over-manned to account for potential shortages. In order to help meet this challenge, the graduates of IMT must be managed more effectively to leverage the Army's fixed manpower to help fill the manning gaps.

In an effort to help streamline the ARFORGEN process, there are some simulation efforts that are ongoing throughout the Army. The ARFORGEN Synchronization Tool (AST) created by FORSCOM is probably the most well-known research effort to streamline the ARFORGEN process. The AST provides an overview of predicted requirements needed across the system as units move through the ARFORGEN process. While AST informs the Army of the requirements that are needed to help streamline ARFORGEN, it does not adjust components such as TRADOC and the way TRADOC offers classes to meet ARFORGEN requirements [23].

USAAC and TRADOC have coordinated with each other to create a simulation that models volunteers entering the Army training system as they move through IMT. While this system does not model ARFORGEN directly, the output of the simulation gives the fill percentage of the newly trained soldiers that are skill level 10 which is assigned to both units following ARFORGEN and the rest of the Army requirements. Thus, by adjusting the controls of the simulation, class schedules can be changed in the model to see the effect the schedules have on the fill percentage of units using the new IMT graduates.

Two types of approaches that could greatly improve the ARFORGEN process rely on simulation and scheduling theory. Simulation has already been used by different organizations as aforementioned. Though the AST is currently unable to adjust the Generating Force structure to adapt to support ARFORGEN, the tool has been helpful in predicting what requirements are needed per unit. Connecting the AST to the model at USAAC and TRADOC have developed could further synchronize the manning requirements by leveraging the new graduates of the IMT.

Scheduling theory is frequently used in various industries such as manufacturing. Through the use of scheduling theory, one can develop appropriate algorithms to determine optimal schedules for various performance measures, which include minimization of makespan, mean flow time, lateness, among others. Scheduling theory can assist TRADOC in adapting to the requirements predicted by the AST. Adjusting TRADOC may require changes in class offerings either by increasing the number or frequency of class offerings so that the soldiers exiting the training system are timed with ARFORGEN requirements.

1.2 Statement of the Army Force Generation Problem (AFGP)

As the operational force transitions to ARFORGEN, the institutional Army must adapt its recruiting and manning processes to support unit requirements in ARFORGEN [26]. Army units such as Brigade Combat Teams (BCTs) are manned based on a modified version of the Headquarters Department of the Army (HQDA) Table of Organization and Equipment (TOE). The Modified TOE or MTOE, provides personnel requirements, where the requirements depend on the type of unit and mission. ARFORGEN generates the modular units such as the BCTs [1]. As the operating force of the Army transforms in accordance with Army guidance, more units are following the progression of the ARFORGEN model. The

generating force, such as TRADOC must adapt to meet the needs of a transforming operating force[28].

The manning of units follows the MTOE authorizations to determine soldiers of required MOS and skill levels. Recently-graduated enlisted soldiers and commissioned officers can satisfy a large portion of a unit's personnel requirements. Aside from new recruits, units are manned with soldiers of more advanced skill levels of particular MOS detailed by the MTOE. The availability of soldiers with more advanced skill levels is subject to whether the soldiers are assigned to another unit, are in training, or on block leave, or are injured to give a few examples. The graphical representation of an enlisted new recruit's flow through the training system and unit assignment is shown in Figure 1.6.

New enlisted and officer recruits join the all-volunteer Army throughout the year. A surge of each group of recruits joins the Army during the months of college and high school graduations. When the new recruits graduate and are ready to be assigned to units, the number of recruits waiting to be assigned reflects the surge of soldiers entering the training system in the summer months. This surge in output of qualified soldiers does not match the modular unit requirements and it leaves soldiers dwelling until a unit enters the Reset/Train phase of ARFORGEN that requires the soldiers. Unlike the previous divisional structure of the Army, the manning of units occurs during one phase called the Reset/Train phase. The newly qualified soldiers need to be scheduled to meet the demands of units entering this first phase of ARFORGEN.

While adjusting the number of volunteers that join the Army during the summer may be a logical fix, the Army cannot control the summer surge to smooth out the graduation of new soldiers to meet ARFORGEN. The factors causing a large increase in recruits in the summer months are entirely external to the Army. Factors such as graduation dates affect the timing of the surge and the state of the economy has an effect on the its size [18].

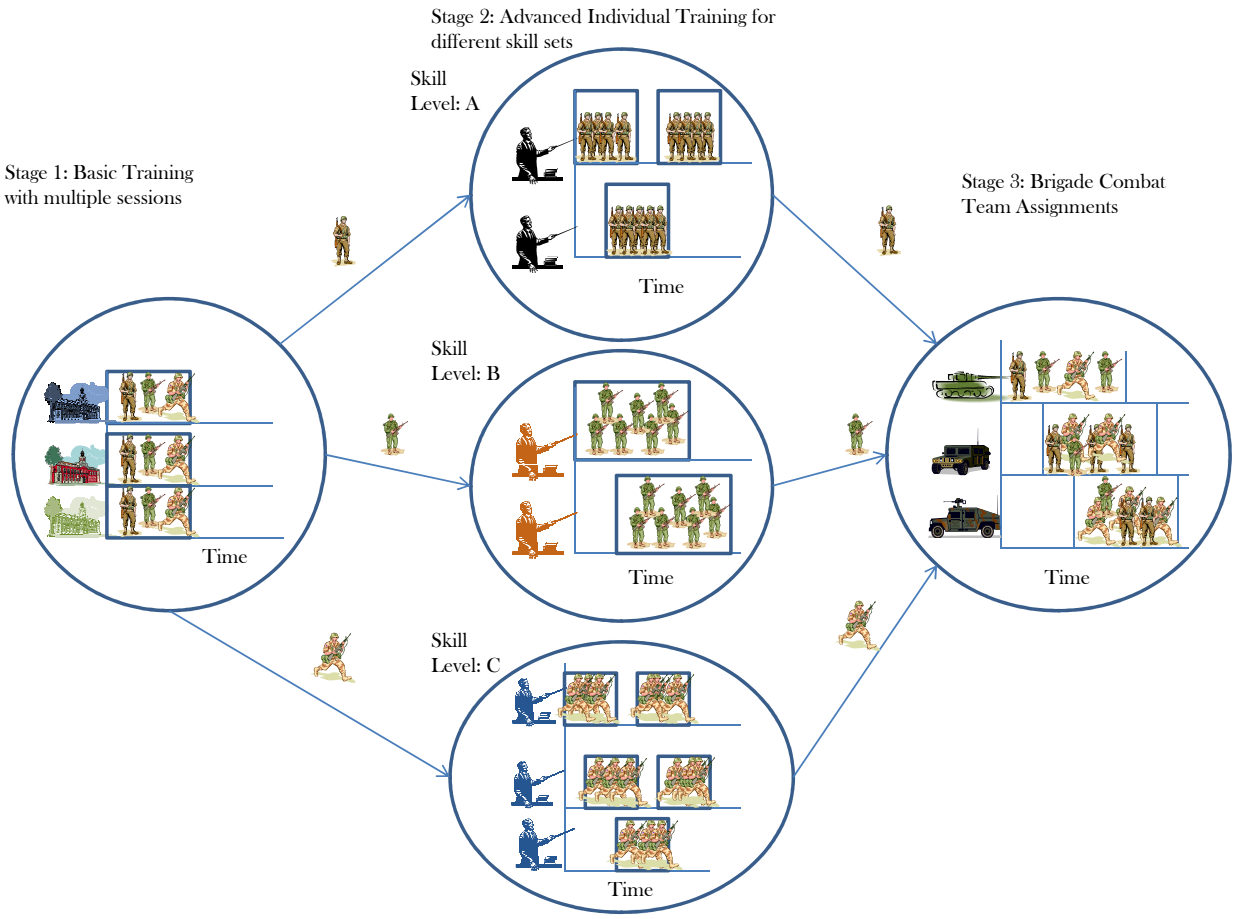


Figure 1.6: Army Initial Military Training in relation to ARFORGEN

However, the flow of the soldiers through the training system can be controlled. TRADOC can adjust class sizes, schedules, and the number of courses offered provided the relative resources can be acquired. Units, such as the Heavy Brigade Combat Team (HBCT) are comprised of various MOSs and ranks of soldiers both enlisted and officers based on the MTOE. In a particular HBCT, newly qualified soldiers account for the majority of the manning requirement where newly qualified soldiers are considered skill level 10 and officers rank as O2. The authorizations for these positions are manned with the soldiers that recently completed their initial training.

Although the Army has both generating and operational force manning requirements, the Army cannot grow bigger than the set end-strength number (the total number of soldiers allowed in the Army including officers, enlisted soldiers and reserves). The Army end-strength number is determined in the yearly-passed National Defense Act. The Army end-strength number is the total number of soldiers the Army is authorized, and it includes all the soldiers that are not assigned to a unit but are part of the generating force, or those that are in training as examples. The Army therefore cannot increase the end strength number if it requires to increase manpower in theater. The Army must instead manage its resources within the set end-strength number. Therefore, a higher number of soldiers in the training system decreases the number of soldiers available in the operating force. One of TRADOC's objectives described by former Commanding General Wallace "is to reduce the time soldiers spend in school while still providing the operating force with highly trained soldiers" [73]. This objective refers to just one aspect that is needed to increase the number of soldiers available to man units. If the soldiers' times spent in schools are reduced but they still dwell before being assigned to a unit, the system would still be inefficient.

The Army requires the system of training soldiers to adapt to the needs of ARFORGEN by producing the right quantity of soldiers with the right skill sets at the right time. By aligning the output of soldiers with the right MOS and AOC to meet unit requirements in ARFORGEN, TRADOC will better meet its requirement to support ARFORGEN by producing a continuous stream of soldiers. In order to link the output with the requirements of units entering the Reset/Train phase, TRADOC may have to offer more classes, more frequently to reduce the mean flow time of each MOS and AOC group moving through the training system. Therefore, the problem that we address in this dissertation can be concisely defined as follows: *Given the arrival times of the enlisted soldiers, determine their assignments to the classes offered in the Basic Combat Training (BT) and Advanced Individual Training (AIT) classes so as to form a required number of Brigade Combat Teams (for AR-*

FORGEN) within a prescribed time window for each and minimize the total flow time of soldiers when the size of a BT class is unlimited while the number and size of AIT classes can vary for a known number of instructors of each skill set. The officers' training problem is a counterpart of this problem except that it does not involve BT classes. The demand for the Army units can be fulfilled late and needs to be accounted for as such. We designate this as the Army Force Generation Problem (AFGP).

1.3 Initial Analysis

In the initial analysis of the training system, the current flow of soldiers through IMT was modeled in a simple numerical example for one MOS to determine the effect of adjusting various parameters on flow time. The initial flow, as detailed in Figure 1.6, shows that the path that enlisted soldiers take includes basic training and advanced individual training before being assigned to a unit. A benchmark model was created based on the way the system currently operates. The number of soldiers that are recruited each month and sent to basic training has an effect on the number of soldiers waiting to start AIT. When the number of soldiers entering the training system is varied using the numerical example, the queue for AIT increased, incidentally causing an increase in flow time. The capacity of AIT classes and schedule were also varied to determine the effect on flow time. When the capacity of AIT increased, the queue between basic training and AIT reduced and flow time was decreased; however, the number of soldiers waiting to be assigned to a unit increased. A variation in the schedule of the AIT classes had an interesting effect on flow time. By simply adjusting the start times of the classes, the flow time could be made worse or better depending on how the start times were adjusted. This initial analysis reveals the scope of obtaining an effective schedule for the training classes.

1.4 Lot Streaming and its Potential Benefits

Lot streaming has been widely researched as a scheduling method to minimize regular performance measures such as makespan, mean flow-time and work-in-progress. Lot streaming is often defined as the process of splitting a job or lot into sublots in a way that allows for concurrent operations [65]. A job in this case is the sum total of sublots that are produced to meet the requirements of some demand. The term job in this case is synonymous with batch since both represent the production order that is followed through a system [76]. When lots are split into sublots, items of a job can move to the next machine when all the items of a sublot finish as opposed to waiting for the entire job to finish processing on one machine. Consider a two machine example where the processing time of machine 1 is 20 units for each item and the processing time of machine 2 is 15 units for each item. In the case without lot streaming a job consisting of a given number of items to process waits for the entire batch to finish each machine.

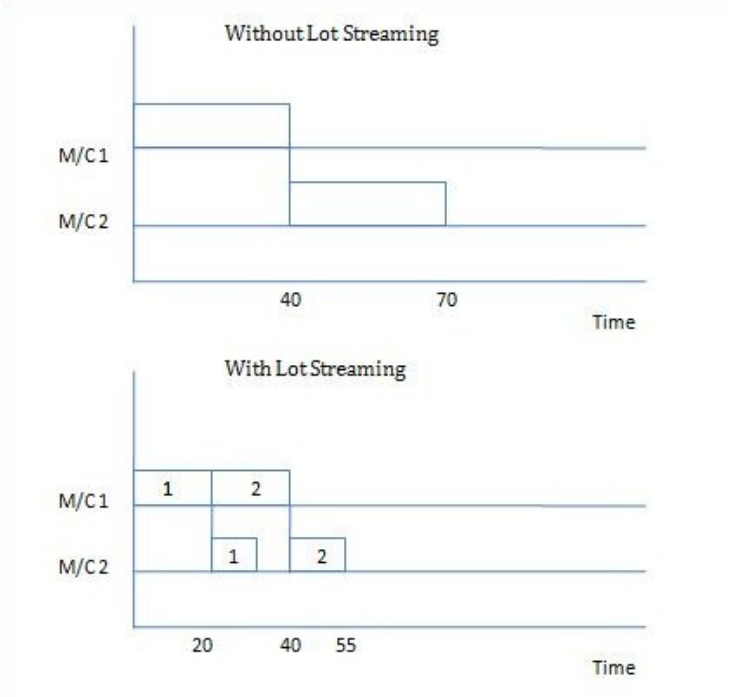


Figure 1.7: No Lot Streaming vs. Lot Streaming Chart.

As shown in the two machine Gantt Chart in Figure 1.7, in the scenario with lot streaming, the makespan is reduced to 55 time units compared to the previous 70 time units. Likewise, the mean flow time is reduced from 55 time units to 37.5 time units.

Other example-based benefits of lot streaming can be found in Kropp and Smunt [49], where the authors describe a 100 unit batch in a three-machine system, with each machine having different processing times. The authors first setup the example without lot streaming as a base case and then, provide comparable results when using lot streaming. The lot streaming results experience a reduction in both makespan and flow-time of 24% and 33% respectively. The authors explain the reduction is caused by the overlapping of operations.

Likewise, Trietsch and Baker [68] also provide examples of lot streaming benefits. The authors describe a scenario such that 84 items require processing on three different machines in a flow shop. They give four different scenarios involving lot streaming to be compared with the case where lot streaming is not used. The examples given are two equal lots with no idling and then with intermittent idling. The first case results in a reduction in makespan of 16.7% and the second case results in a reduction in makespan of 25%. The other two examples described use varying lot sizes first with no idling and then with intermittent idling. The reductions on makespan in the two cases compared with the base case are 23.6% and 28.6%, respectively. Similar to the case of equal lots, the reduction in makespan is greater when intermittent idling is allowed.

Similar to the example depicted in Figure 1.7, benefits of lot streaming can be shown graphically with the use of a Gantt chart or described through the numerical reductions of makespan or flow-time as presented in [49] and [68]. Besides the aforementioned examples, simulation is also a popular tool to use when reporting the benefits of lot streaming. However, analytical results on the potential benefits of lot streaming have been presented by Kalir and Sarin [46]. They provide these results for the three common performance measures: makespan, mean flow time and work-in-progress. Given the problem parameters, the

analytic results of Kalir and Sarin [46] show that lot streaming reduces the three performance measures in the case of single and multiple lots when setup and transfer times are considered negligible. The authors present upper bounds on the benefits of lot streaming for both single and multiple lot cases.

1.5 Use of Lot Streaming for the AFGP

To motivate the discussion for the use of lot streaming in the context of ARFORGEN problem, consider the manning problem the Army faces. During a given period of time the Army has to draw on the available operating force to fill Army units, battalions, and fulfill other requirements based on the force structure requirements set in the Modified Table of Organization and Equipment. The personnel requirements list the skill level and rank of a soldier in the quantity set by the MTOE. The force structures of the units require a mixture of both officers and enlisted soldiers of particular ranks and occupational skill sets. For example, four 11B at skill level 10 may be required as well as three 11B at skill level 20. Some Battalions and all Brigade Combat Teams follow a deployment cycle according to the Army Force Generation. During the first phase of the ARFORGEN cycle, the Reset/Train phase, the Army has a specified window of time, usually six months, to man the units according to the MTOE requirements. A range of three to ten BCTs and battalions could enter the Reset/Train phase of ARFORGEN within one month, every month. The Army has six months to standup each unit, starting from the time the unit enters the Reset/Train phase. The personnel available to meet the requirements of the units entering the Reset/Train phase are soldiers just graduating from Initial Military Training (IMT), soldiers finishing training beyond initial skill sets, soldiers leaving other units after promotions or because of reassignments, and some soldiers may stay with the unit.

A Soldier's graduation from IMT depends on the advanced training of his particular

skill set. After the completion of basic training, soldiers attend different schools with different training requirements and course lengths. Therefore, volunteers entering the Army at the same time may not finish at the same time. Officers and enlisted soldiers move through separate training systems managed by Training and Doctrine Command (TRADOC). Enlisted soldiers attend advanced individual training specific to their military occupational speciality. Every MOS has its own training requirements for qualification of a particular skill level. The enlisted soldiers must attend training at Basic Combat Training prior to AIT. No matter the MOS, all soldiers must attend this training to begin the Soldierization process. Depending on resources, multiple sessions of BCT could be occurring concurrently and at different facilities. One exception to the BCT to AIT flow is what is called One-Unit Station Training (OSUT). There are particular MOS where BCT and AIT occurs at the same place and as one long training class. Prior to starting BCT, enlisted members are tested and qualified for their desired MOSs. United States Army Accessions Command oversees this aspect along with the recruitment.

The officers attend training classes specific to their Area of Concentration (AOC) during Basic Officer Leadership Course III. The training the officers and enlisted soldiers attend has a capacity based on teacher to student ratios. Depending on instructor availability and density of MOS or AOC, training classes could be offered concurrently. When both the officers and enlisted soldiers complete training, they wait for the Human Resource Command (HRC) to assign them to a unit. Given the number of soldiers required to meet unit demands during a given month, the output of soldiers from the IMT provides a large source of skill level 10 MOS and junior officers. The skill level 10 MOS specifically make up a large portion of the MTOE requirements of Basic Combat Teams. If this resource stream is sequenced to meet appropriate unit demands by the end of the Reset/Train phase following ARFORGEN and the demands of the Army not following ARFORGEN, while minimizing the mean flow time of groups of MOS and AOC, the Army could leverage its resources against (fixed)

end-strength so that more soldiers are available to meet the Army's personnel requirements.

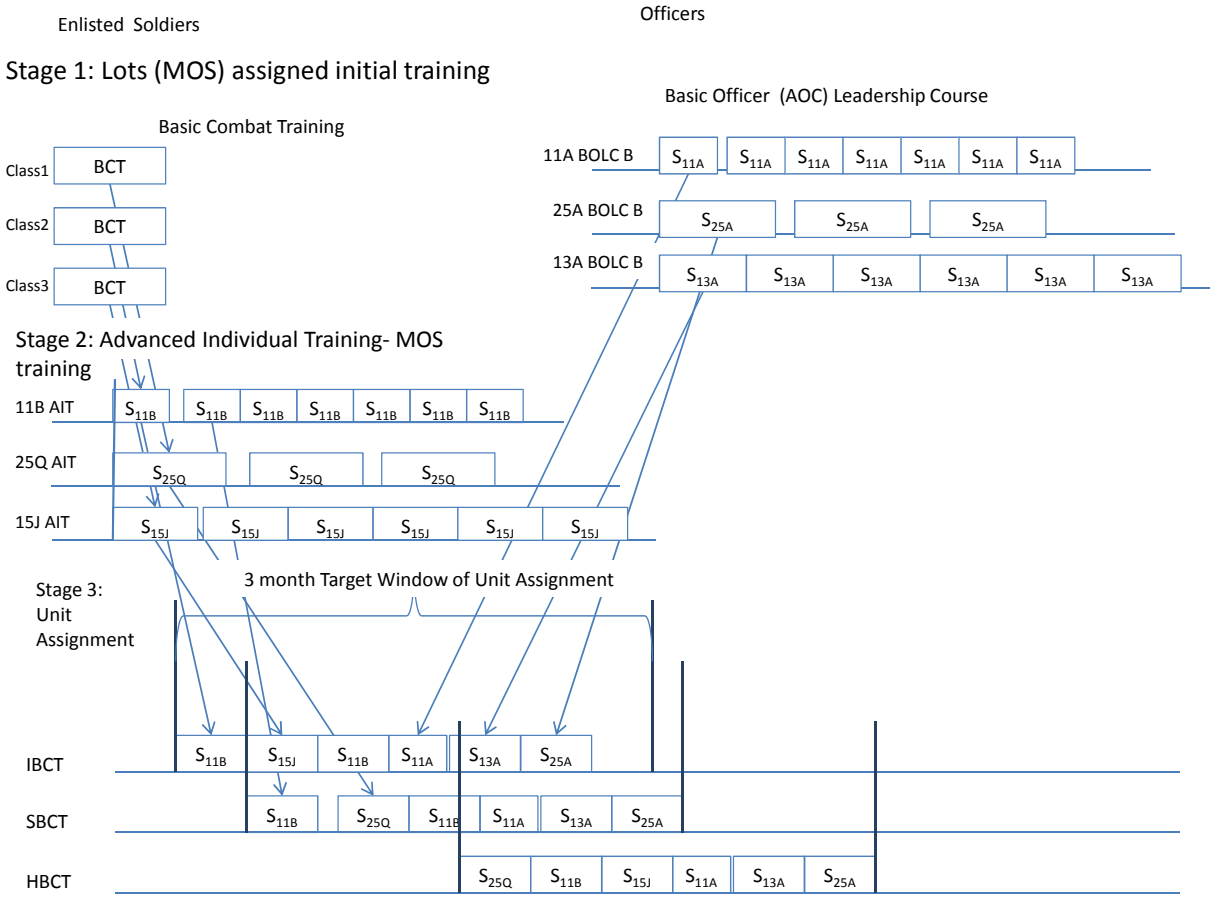


Figure 1.8: An example of lot streaming applied to the Initial Military Training

Applying the lot streaming concept to this system, Figure 1.8 shows soldiers moving through the system in lots that are based on soldiers' MOS or AOC. Each lot is divided into smaller batches, or sublots to begin their transition through the system based on the demand of their skills. Starting at Stage 1 in the figure, each class represents a concurrent session of Basic Combat Training. Each box that has "S" is a sub-lot. The subscripts of S_{11B} (for example) represent skill set number and the field that it is for MOS or AOC. The width of the box represents the time of the course. At the end of the box, the sublot will move to

Stage 2: Advanced Individual Training for the enlisted scenario. In Stage 2, the widths of the boxes are different for each MOS to represent different lengths of courses depending on MOS. The completion of these courses then have an effect on the availability of the soldiers for the units in Stage 3. Stage 3 represents the Basic Combat Teams in the Reset/Train phase of ARFORGEN, where units must fulfill their personnel requirements. A three-month target window is preferential to taking the full six months as shown in Figure 1.8.

Therefore, the formation of Basic Combat Teams (BCT) is a lot streaming type of problem. When volunteers enter the Army in both the enlisted and officer systems, they move to a form of advanced training (stage two for the enlisted system and stage one for the officer system). As enlisted soldiers and officers move to their advanced training courses, they attend a training program specific to their MOS or AOC, respectively. As the collection of identical MOS or AOC are moved to advanced training, it is split into smaller lots based on the limited capacity of training, that is, the number of instructors, size of classrooms, and availability of a facilities, to name a few. The classes then stream concurrently depending on the number of instructors. Thus, the questions that lot streaming research may help solve are what are: the optimal classroom sizes, and how should the classes be streamed to minimize mean flow time.

Lot streaming is an attractive scheduling tool to use given its proven benefits to performance measures such as makespan, mean flow time and average work-in-progress level [65]. The performance measure most appropriate for the problem on hand is mean flow time (MFT). By minimizing the average time a soldier spends in the system, the availability of a soldier increases. When a soldier moves through the training system, he is not considered part of the operating force, that is, he is not considered part of the soldiers available for unit assignment or deployment. While this soldier remains in the training system, he receives full pay and benefits to the cost of the Army. With the current schedule, soldiers often wait for long periods of time for an available training seat in the MOS and AOC-specific

training classes. Minimizing the time soldiers spend in the training system will also reduce the waiting time. Minimizing the time in system will reduce the manpower cost by increasing the Army's ability to leverage new graduates to meet unit manpower gaps and reduce the financial cost associated with soldiers in the training system.

Next, we study potential benefits accrued from applying lot streaming to the formation of units by comparing the mean flow time obtained under lot streaming (designated as MFT^{LS}) and the MFT obtained for the no lot streaming case as shown in [65].

Notation.

The following notation is used in the calculations of the potential benefits when using lot streaming.

Parameters

ES	Total number of enlisted soldiers
O	Total number of officers
$P^{(1)}$	Processing time of Basic Training
$I^{(2)}$	Total number of instructors in AIT for the enlisted training system
$P^{(2)}$	Processing time of AIT
$OP^{(1)}$	Processing time of the Basic Officer Leadership Course
$OI^{(1)}$	Total number of instructors in BOLC B for the officer training system
p_l	Processing time of machine l where each machine represents a stage in the training system

Variables

$B^{(1)}$	Total number of sublots in Stage 1 for the enlisted training system
$B^{(2)}$	Total number of sublots in Stage 2 for the enlisted training system
$OB^{(1)}$	Total number of sublots in Stage 1 for the officer training system
$s_i^{(1)}$	Sublot size of sublot i in Stage 1 for the enlisted training system
$s_{ka}^{(2)}$	Sublot size of sublot k on machine a in Stage 2 for the enlisted training system
$Os_{ej}^{(1)}$	Sublot size of sublot e on machine j in Stage 1 for the officer training system

Consider the MFT required in Stage 2 when lot streaming is not used (as shown in [65, page 12]):

$$MFT = ES \sum_{l=1}^{I^{(2)}} p_l (= ESP^{(2)}). \tag{1.1}$$

Similarly, MFT for Stage 1= $ESP^{(1)}$.

Now, consider the MFT^{LS} by first observing that the flow time with lot streaming is designated as FT^{LS}.

$$FT^{LS} = \sum_{i=1}^{B^{(1)}} s_i^{(1)} P^{(1)} + \sum_{a=1}^{I^{(2)}} \sum_{k=1}^{B^{(2)}} s_{ka}^{(2)} P^{(2)} + \sum_{j=1}^{OI^{(1)}} \sum_{e=1}^{OB^{(1)}} Os_{ej}^{(1)} OP^{(1)}. \tag{1.2}$$

The first two summations are the flow times for sublots of the two stages in the enlisted training system. The third summation is the flow times of the sublots in the officer training system. The assignments to units at the last stage for both the systems require no processing time, and is, thus, excluded in the representation of MFT^{LS} and MFT. We have, the MFT^{LS} as follows:

$$MFT^{LS} = \frac{1}{U}\left(\sum_{i=1}^{B^{(1)}} s_i^{(1)} P^{(1)} + \sum_{a=1}^{I^{(2)}} \sum_{k=1}^{B^{(2)}} s_{ka}^{(2)} P^{(2)}\right) + \frac{1}{O}\left(\sum_{j=1}^{OI^{(1)}} \sum_{e=1}^{OB^{(1)}} Os_{ej}^{(1)} OP^{(1)}\right). \tag{1.3}$$

Noting that the sum of the sublots must equal the total number of soldiers in the system, the MFT^{LS} can be written as

$$\begin{aligned} MFT^{LS} &= \frac{1}{ES}\left(P^{(1)}(s_1^{(1)} + s_2^{(1)} + s_3^{(1)} + ... + s_{B^{(1)}}^{(1)}) + P^{(2)}(s_{1,1}^{(2)} + ...s_{B^{(2)},1}^{(2)} + ... + s_{1,I^{(2)}}^{(2)} + ... + s_{B^{(2)},I^{(2)}}^{(2)})\right) \\ &\quad + \frac{1}{O}\left(OP^{(1)}(Os_{1,1}^{(1)} + ... + Os_{OB^{(1)},1}^{(1)} + ... + Os_{1,I^{(2)}}^{(1)} + ... + Os_{(OB^{(1)},I^{(2)})}^{(1)}\right) \\ &= \frac{1}{ES}\left(P^{(1)}(ES) + P^{(2)}(ES)\right) + \frac{1}{O}\left(OP^{(1)}(O)\right) \\ &= P^{(1)} + P^{(2)} + OP^{(1)}. \end{aligned} \tag{1.4}$$

The MFT for the enlisted soldiers and officers can also be written as $MFT = ES(P^{(1)} + P^{(2)}) + O(OP^{(1)})$. Therefore, we have

$$\begin{aligned} \frac{MFT^{LS}}{MFT} &= \frac{P^{(1)} + P^{(2)} + OP^{(1)}}{ES(P^{(1)} + P^{(2)}) + O(OP^{(1)}} \\ &\leq 1. \end{aligned} \tag{1.5}$$

Hence, under lot streaming, the MFT can be improved as ES and $O > 1$.

A reduction in flow time of each sublot reduces the lead time, thereby enabling soldiers of necessary skill sets to join appropriate Units by their due dates that are specified by the end of the Reset/Train windows. Hence, minimizing mean flow time is an appropriate measure for the problem on hand. Besides getting the soldiers with the right skill sets to the Units in shorter lead time, a shorter flow time reduces idle time of soldiers and the manpower cost to the Army. As each sublot is in the training system, the Army pays each soldier's salary whether the soldier is attending class or waiting for a class to begin.

Chapter 2

Literature Review on the use of Lot Streaming in Various Machine Configurations

Lot streaming problems, like any other scheduling problem, can vary from one another because of the presence of different features. Sarin and Jaipraskash [65] use an eight field classification scheme, expanding on the scheme of Trietsch and Baker's [68] to classify different types of lot streaming problems. This eight field classification scheme is as follows [65].

- **Number of machines and configuration:** The number of machines in the problem represented by m, arranged in either Flow Shop (F), Job Shop (J) or in Parallel (P) [65].
- **Number of lots:** The number of lots or jobs a problem has to process. A lot is a group of identical jobs [68].
- **Sublot type:** The size of the sublots can be equal, consistent or variable. For consistent sublots, the sizes of sublots remain the same throughout processing but they are not necessarily equal to each other. The equal sublot size case is a special case of consistent sublots. Even though enforcing equality of sublots does not affect small-sized

problems, the equal sublot restriction on large-sized problems even when sublots were previously consistent, may increase the makespan of a problem (Trietsch and Baker [68]). Sublots can also be variable, in which case neither of the equality or consistent restrictions holds. Variable sublots allow a sublot to be of a different size for processing at different stages of production. Hence, when sublots re-form between stages, a new sublot can be made from a combination of multiple sublots from the previous stage. The advantages of variable sublots on optimal makespan are discussed in [7].

- **Idling|Non-Idling:** As the sublots are processed by a machine, the problem can be classified as non-idling if the sublots are required to be completed continuously after each other with no delay. If the sublots do not have to be completed continuously after each other on a machine, the problem can be classified to have intermittent idling [7].

- **Sublot Size:** The size of a sublot can be restricted to be an integer (for discrete sublot sizes). Otherwise, the sublot size can be continuous.

- **Objective Function:** This field classifies the problem in accordance with the objective function, which can be minimizing the makespan such as flow time, mean flow time or work-in-progress. A typical objective function used in the literature for lot streaming problems is minimizing the total makespan.

- **Special Features:** Some additional relevant features for lot streaming problems can have are attached-setups, detached-setups or no setups, intermingling of the sublots, item or sublot availability, and no-wait or wait schedules to name a few. In a setting with an attached-setup, the setup cannot be performed until the sublot arrives at the machine. For the detached-setup case, the setup can start while the sublot is still being processed on the previous machine. There are some cases where the setup time is either negligible or does not exist at all, which constitute the no setup case [7]. Some cases of lot streaming include the intermingling of sublots from different lots [65]. If a sublot is transferred to the next stage on an item by item basis, it is termed as "item

availability," otherwise if a sublot is transferred to the next stage only when the entire sublot is completed, it is called "sublot availability." The no-wait schedule requires the sublots to be processed on the next stage without any delay from its processing on the previous machine. The wait schedule allows a delay in the processing of a sublot on the next stage after its completion on the previous machine (stage) [7].

2.1 Flow Shop

2.1.1 Two-Machine

Using the classification structure, consider the $2(F)/n/\{NI, I\}/\{DV, CV\}/\{C_{max}, \sum C_j\}$ or two machine set of problems. Potts and Baker [59] have considered 2-machine, minimizing makespan problem for the case of consistent sublots. They compare the optimal makespans obtained using equal sublots and consistent sublots, and show that "$M^E/M^C < 1.09$" where M^E is the makespan for equal sublots and M^C is the makespan for consistent sublots. This bound value is 1.53 when $m = 3$.

Trietsch and Baker [68] have addressed the two machine $2(F)/n/C/NI/C_{max}$ problem for both the continuous and discrete versions. They show that: (i) solving the $2/C/NI$ model finds the minimum makespan for the two machine problem; (ii) there is no reasonable use of variable sublots in the two machine scenario since only one transfer between machines occurs; and (iii) there is no need to allow intermittent idling since running either machine continuously has no effect on the minimal makespan. They also use the observation that the optimal sublot sizes are geometric to provide a comparison between optimal continuous solutions and discrete solutions. They present a polynomial-time procedure that avoids use of the integer programming, to find integer optimal solutions.

Baker [2] has addressed a 2-machine lot streaming problem for the objective of mini-

mizing the makespan in the presence of lot-attached/detached setups, and used a sequencing method that relies on the use of Johnson's rule. Vickson [72] has extended the use of lot streaming to accommodate multiple products or lots, and has presented closed-form solutions to the lot-streaming two machine flow shop problem when setup times and sublot transfer times are considered. The setup times cover both lot-detached and lot-attached setups.

Bukchin, Tzur and Jaffe [9] have addressed minimizing average flow time as the objective function for a 2-machine lot streaming problem with special features such as sublot-attached setup times, variable processing times and sublot setup times that are machine specific. Bukchin *et al.* [9] use insights from Single Machine Bottleneck characteristics of optimal solutions to guide their solutions procedure. An interesting result is the comparison of the makespan and average flow time results. Bukchin *et al.* report that when the optimal makespan solution is used for the average flow time objective, the average flow time is naturally not optimal; however, interestingly the differences in the optimal average flow time solution and solution of average flow time when makespan is the objective are very large. On the other hand, the solution for the average flow time as the objective results in smaller differences than the reverse.

2.1.2 Three-Machine

Continuing the classification structure, consider the

$3(F)/n/\{C, E, V\}/\{NI, II\}/\{DV, CV\}/\{C_{max}, \sum C_j\}$, or three machine set of problems. The 3-machine problem is researched less often than the 2-machine and m-machine lot streaming problems. Three-machine problems provide a steppingstone for the m-machine problem. Trietsch and Baker [68] use a linear programming formulation to find the optimal consistent sublot sizes. They present two linear programs from different perspectives. Both linear programs presented minimize makespan, although one does so explicitly through com-

pletion times. The second model uses total idle times and total processing time on the last machine as an implicit way to minimize completion time. They also addressed the $3/V/NI$ problem. They create a partition set of machines that run continuously in order to have conditions for no-wait schedules. With the conditions for no-wait schedules, they are able to construct a set of sublots whose sizes are geometric.

Baker and Jia [4] compared the performances of different lot streaming problems. They addressed the 3-machine problem and the effect the constraints, appropriate for the alternate lot streaming procedures, have on makespan. By using a testbed of problems, they were able to compare equal, consistent, and variable sublots for both no idling and idling problems. They report that the additional constraints have an impact of the order of $5-10\%$ on the makespan compared to the problem without the more constrictive constraints. This impact is different in the cases where machine 2 is dominant and when machine 2 is dominated. When machine 2 is dominated, the use of consistent or equal sublots impacts the solution by at least an estimated 10%. When the no idling constraint is added, the solution is impacted by at least 50%. Furthermore, by adding these constraints, a problem may become easy to solve. This analysis gives insight into the sub-optimality costs of adding the additional constraints.

Glass, Gupta and Potts [33] addressed the $3(F,J,O)/1/C/CV/C_{max}$ lot streaming problem with no setup times. They approach the lot streaming problem by using network notation to determine optimal sublot sizes. By using the relationships $p_2^2 < p_1p_3$, $p_2^2 = p_1p_3$, and $p_2^2 > p_1p_3$ where p_i for $i = 1, 2, 3$ are the process times on the respective machines, i, critical path can be determined of the network for the three different processing time cases which gives corresponding optimal sublot sizes. Results based on the respective relationships of the machine processing times are presented that calculate optimal sublot sizes. The parameters of the calculations are solely based on the machine processing times while the number of sublots is given.

Chen and Steiner [17] extend the work of Glass *et al.* on the $3(F)/1/C/CV/C_{max}$ problem by taking into consideration setup times as a special feature. They focus on lot-detached setup times and present a formula for calculating the optimal makespan for such a case. They represent the problem as a network similar to that in Glass *et al.* [33]. The calculation of the optimal sublot sizes is based on the relationship of the processing times used by Glass *et al.* [33]. The additional consideration of setup times is accounted for with an index of pattern-changing sublot to help determine the optimal sublot sizes. Continuing with their analysis of detached-setup times, Chen and Steiner [17] focused on the attached-setup times lot streaming case. Instead of presenting a formula to calculate the optimal sublot sizes, they present several different cases of the problem with lot-attached setups to find optimal sublot sizes that minimize makespan.

2.1.3 m-Machine

The m-machine lot streaming problem or $m(F)/n/\{E,C,V\}/\{CV,DV\}/\{C_{max},\sum C_j\}$ has been extensively researched as the lot streaming research has grown. Naturally, a viable approach is to adjust successful procedures for the case of $m \leq 3$, and apply them to the general case. Baker and Pyke [3] have presented a simple algorithm to determine an optimal solution for the two-sublot $m(F)/1/\{E,C\}/CV/C_{max}$ lot streaming problem. The algorithm finds the first critical machine, and with a critical coefficient, the size of the sublot is calculated using the coefficient and the geometric relationship among optimal sublots. They have adapted this algorithm for the $m(F)/n/\{E,C\}/CV/C_{max}$ problem by using the same coefficient found in the 2-sublot solution and again using the geometric nature of optimal sublots to determine n-sublots. They report that the n-sublot heuristic results in near-optimal solutions.

Kropp and Smunt [49] have addressed the $m(F)/1/E/CV/\sum C_j/$*Detached-Setup Time*

lot streaming problem at a time when the makespan was primarily the focus. Incidentally, Kropp and Smunt [49], provide models that minimize either makespan or mean flow time. The use the objective function of minimizing the mean flow time that includes cross-products and squared terms, and therefore, it leads to a quadratic programming problem. They briefly discussed the integer sublot size version of the problem. They claim that the continuous version of a problem with rounding to an integer solution, while preserving feasibility, is a reasonable approach for larger problems, that is, when the total number of items in the system is large. The relationship between the setup times and mean flow time is analyzed to better understand the properties of a heuristic that minimizes mean flow time. They report a linear relationship between mean flow time and setup-to-processing time ratio, which supports claims that flow time can be reduced by decreasing setup time. This relationship also reveals how the economic ordering quantity model underestimates the value of reducing setup times to minimize flow time. They have also analyzed the effect various setup times have on sublot size distribution, and found unequal sublot sizes to be optimal when setup times are minimal or close to zero. As the setup times increase, they report that the optimal sublot sizes become more equal. They also found that in the presence of setup times, the first sublot was very small. This first sublot acts, as what they call, a flag for machines down the line to setup. This observation is the basis of Kropp and Smunt's flag heuristic. The flag heuristic splits the lot so that the first sublot is the smallest feasible size and the other sublots are equal in size. They report this heuristic approaches optimality when the ratio of setup-to-processing time grows. Thus, the heuristic works well for moderate to high setup times for the deterministic, single-job flow shop case.

Glass and Potts [33] extend the analysis of the 3-machine problem to the m-machine problem, $m(F)/1/C/CV/C_{max}$. They approach the m-machine problem by investigating dominance properties of the machines such that some machines can be eliminated from consideration. They also use a network representation of sublot sizes similar to that in [33]

with the aim of finding the optimal sublot sizes by selecting the best nodes such that the critical path is minimized. After first reducing the number of machines by using a Relaxation Algorithm and setting up the critical path structure of an optimal solution, they customize this structure for a lot streaming problem. The structure for a lot streaming problem takes the form of a critical block with optimal sublot sizes. They report this algorithm to be more efficient than the linear programming counterpart for a small number of dominant m-machines (when $m = 3$ or $m = 4$).

Ramasesh *et al.* [62] investigated a multi-stage manufacturing system, which is similar to an m-machine flow shop problem. They determined optimal lot sizes for a single lot problem for the objective of minimizing the work-in-progress costs. Using lot streaming and lot sizing literature, they developed an economic production model to determine optimal sublot sizes that minimize inventory cost.The resulting model shows significant reduction in total cost when compared with the lot sizing model without lot streaming.

Kalir and Sarin [44] have presented an efficient algorithm to solve the $m(F)/1/E/C_{max}/$ *Sublot-attached Setup times* lot streaming problem. They give an expression for finding the bottleneck machine while accounting for sublot-attached setup times. In addition, they present a closed-form expression for finding optimal sublot sizes. Some of the machines in the m-machine problem can be eliminated as a result of the "Dominance Property" that they present. By using this property and expression for sublot sizes, which accounts for sublot-attached setup times, Kalir and Sarin present an algorithm that solves the stated problem in $O(m)$ computations. They also report, that since the algorithm does not depend on the number of sublots in a problem, this algorithm can thus address larger problems efficiently.

Extending this research to the n lot problem, Kalir and Sarin [47] address the $m(F)/n/E/C_{max}/$ lot streaming problem without sublot-attached setup times. They use the observation that lot streaming sequencing problems with small sublot sizes result in

near-optimal solutions when idleness is minimized, to develop a heuristic. Their heuristic, named, bottleneck minimal idleness (BMI), heuristic uses this insight to develop a schedule that minimizes idleness on the bottleneck machine. They present results of experimental investigation that show that the BMI heuristic outperforms the commonly used fast insertion heuristic (FIH). Unlike the FIH method, the BMI method is not as sensitive to either the level of dominance or location of a machine.

Kalir and Sarin [45] have addressed the single and multiple-batch flow shop lot streaming problems with sublot-attached setup times. They have developed an optimal solution algorithm of low polynomial order for the single-batch problem. They present several algorithms for the multiple-batch problem for when the sublot sizes are identical and product-based. For the two-machine flow shop lot streaming problem, they present an optimal pseudo-polynomial solution using Johnson's rule. Using the solution algorithms they present for the two-machine flow shop lot streaming problem, Kalir and Sarin extend their algorithm to solve the m-machine flow shop sequencing problems. Since Johnson's rule yields optimal sequences for the two-machine flow shop, their solution method for the m-machine flow shop is a near-optimal heuristic.

Most of the previous papers reviewed thus far have considered either equal or consistent sublots as a solution to a lot streaming problem. Liu [54] has presented a heuristic to solve the $m(F)/1/V/DV/C_{max}$ lot streaming problem. The heuristic consists of only two steps. The first step is to find the continuous solution to a discrete solution. In order to find the continuous solution, Liu [54] used the relaxation algorithm and dominant machines presented in [34]. Using this method to determine the dominant machines and critical blocks, sublot sizes are allocated for each critical block by an allocation ratio. Adjacent sublots are equal to each other when the allocation ratio is multiplied in each critical block. Thus, flexible sublot sizes are allowed. Given those optimal variable sublot sizes, a linear program is used to determine the makespan. This method is reported to perform better than other

discrete lot streaming models since it results in the lowest average deviation percentage from the best solution.

Biskup and Feldmann [7] have presented research for the discrete, variable sublot size problem. Specifically, the problem is classified as $m(F)/1/II/V/DV/C_{max}$. Biskup and Feldmann [7] are the first to present an integer programming formulation for the discrete, variable sublot problem. In the formulation, they use binary variables to ensure that sublots cannot be formed on a machine before items on the previous machine are completed. To check the degree by which variable sublots improve the makespan, the model was run using equal and consistent sublots. They report an improvement of up to 12.7% over equal sublots and 3.5% over consistent sublots when compared with the use of variable sublots. They have also extended their formulation to account for attached- and detached-setup times through additional constraints that can be added to the formulation.

Sarin, Kalir and Chen [66] have addressed determining the number of sublots of a single-lot, multiple-machine flow shop lot streaming problem in order to minimize a unified cost-based objective function including multiple performance measures such as makespan, mean flow time, work-in-process, sublot-attached setup and transfer times. They present a polynomial-time procedure to determine the number of sublots. Through computational experiments they have discovered their solution procedure very often finds an optimal solution.

Feldmann and Biskup [32] have investigated the m-machine, n-lot problem with intermingling allowed. They have presented another formulation that allows for intermingling as well as other variations. When intermingling is allowed among sublots, the sequence of sublots of a lot can be interrupted by another sublot of another lot. Two extensions are provided for the intermingling model. One extension permits two sets of sublots one that allows intermingling and one set that has one or more sublots that are not allowed to intermingle. The additions to the model with this extension is the use of a binary variable that accounts

for the sequence. The other extension accounts for the situation where the overall number of sublots is given, but the number of sublots per product is not given. A binary variable is used again, but in this case, the variable accounts for the position at which a product is produced instead of the sequence. The formulation that accounts for the sequence requires far fewer iterations than the position-binary variable. The formulation using the sequence binary variable is used to solve 160 instances generated from a problem generator they call LSGen. In one example, they report a reduction in makespan when intermingling is allowed over when intermingling is not used. From the generated problem results, the marginal benefits are reported for using consistent intermingling sublots. Based on the results presented, the benefits of lot streaming are increased as the number of stages and sublots increase.

Marimuthu *et al.* [55] use a genetic algorithm (GA) and a hybrid evolutionary algorithm (HEA) to approach the $m(F)/n/NI/E/DV/\{C_{max}, \sum C_j\}$ lot streaming problem. They compare their GA and HEA to what they refer to as Baker's Algorithm and simulated annealing over randomly generated test problems. The GA and HEA are reported to optimize the makespan and total flow time as compared to Baker's Algorithm and simulated annealing. They also claim that the GA and HEA can provide solutions to any scheduling performance measure such as minimum makespan, flowtime, tardiness, cost, mean flow time, mean weighted absolute deviation from due dates as well as other performance measures.

Martin [56] has also used a GA approach for the $m(F)/n/NI/C/DV/C_{max}/Setup\ Time$ lot streaming problem. Using the GA formulation, the number of sublots and their sequence are determined and then, sublot sizes are determined using a mathematical program, which can be easily modified to consider various performance measures. Since the GA determines the number of sublots, the problem can be extended to solve the instance when the number of sublots is variable, though with more computational effort.

Bukchin, Masin and Kirshner [10] have addressed a multi-objective lot streaming

problem. The two objectives considered are makespan and flowtime, which tend to be the most common in lot streaming research. They look specifically at the following problem:

$m(F)/n/C/DV/\{C_{max}\&\sum C_j\}/Setup\ Time$, problem where the number of sublots are unknown. They developed an efficient frontier approach using what they refer to as diversity maximization approach (DMA) coupled with a mixed integer programming formulation to address the multi-objective problem in order to generate a set of preferred solutions. After establishing a set of preferred solutions, the most-balanced-solution is applied to find a solution that is optimal for both makespan and flow time objectives. Using this approach to solve the multi-objective problem, minimizing both makespan and flow time results on the average, of a deviation of about 4.6% and 2.5% from the furthest and closet objective.

Defersha and Chen [21] have extended on previously discussed works of Biskup and Feldmann, [7] and [32]. Specifically, they addressed a multi-product instead of the single product variable sublots lot streaming problem in the presence of setups. A basic model is presented that determines the number and size of sublots for each product and the sequence of processing the sublots while minimizing the makespan. The basic model is extended to account for attached and detached extensions of the model with the addition of a few constraints, and solved using a combination of a genetic algorithm and the simplex algorithm through ILOG CPLEX. By using their hybrid genetic algorithm, Defersha and Chen are able to quickly generate solutions to the computationally burdensome variable sublot-size problem.

2.2 Other Machine Configurations

Lot streaming has also been applied to machine configurations besides flow shop, as we present next.

2.2.1 Hybrid Flow Shop

Zhang *et al.* [76] investigate the $m(H)/n/NI/E/DV/\frac{1}{n}\sum_{k=1}^{n} C_{max_k}/Setup\ Time$ problem, where the (H) represents hybrid flow shop. Essentially in this case of a hybrid flow shop, there is a set of identical or parallel machines available in one or more stages. Specifically, Zhang *et al.* investigated a two-stage hybrid flow shop in which m identical machines are in the first stage and a single machine is in the second stage. They approached the problem by developing two heuristics each of which first sequences the jobs, and then, schedules them using lot streaming. Linear Programming (LP) is also used to determine sublot sizes. In the first heuristic, called "whole-job sequencing," the jobs are sequenced before the lots are split. The second heuristic uses "aggregated-machine sequencing". Instead of first sequencing the jobs followed by lot streaming as in the first heuristic, the second heuristic solves the single-job lot streaming problem for each job using an LP, determines the time requirements, and then sequences the jobs accordingly based on the time requirements. The performances of these two heuristics, are compared based on a lower bound on the optimal solution. A computational investigation is presented to show that, the second heuristic performs better than the first heuristic. The solution obtained by the second heuristic is only shown to be 6.85% away from the lower bound on average.

Liu [53] has considered lot streaming in a single-job hybrid flow shop environment with m identical machines at the first stage and a single machine at the second stage in order to minimize the maximum completion time, similar to Zhang *et al.* [76]. Liu also uses a sequencing algorithm followed by a linear program to determine sublot sizes. However, a "rotation" method is used to sequence the sublots on m identical machines in the first stage. The sublots are assigned across the machines so that each machine has at least one sublot assigned before the $m + 1^{th}$ sublot is assigned to the first machine. The "rotation" allocation of sublots is shown to generate an optimal solution. Also, heuristic methods are presented. In one heuristic, where equal sublots are considered, the problem is shown to

be solved efficiently if the number of sublots is known. The second heuristic, builds on the results of the first heuristic, and sublot sizes are optimized by using a LP model.

Cetinkaya and Duman [15] consider a hybrid flow shop consisting of a mixed flow shop and open shop at each stage. Specifically, they address a 2-machine, two-job lot streaming problem for the objective of minimizing the makespan. Under specific conditions, it is shown that lot streaming can improve the makespan in a mixed shop by sequencing the flow shop jobs before the open shop jobs in the first stage, and then, sequencing the flow shop jobs after the open shop jobs in the second stage, in accordance with the Johnson's rule [15].

2.2.2 Parallel Machines

Kim, Kang and Lee [48] address the two-stage flow shop problem with identical parallel machines at each stage, which could also be viewed as a flexible flow shop. They consider n jobs with transfer batches between the stages and setups separated from processing and not separated from processing in finding the permutation schedule that minimizes makespan. In both cases the concept of the run-in and run-out delays of the jobs is used to sequence the jobs. They show that the scheduling rule that uses these rules to sequence the jobs produces an optimal solution, and it is similar to Johnson's rule [15].

Lin and Jeng [51] have discussed a parallel machine problem with batch scheduling in order to minimize the maximum lateness and the number of tardy jobs. They develop two dynamic programming algorithms in order to find an optimal solutions for two different objective functions, namely, minimize maximum lateness and number of tardy jobs. The dynamic programming algorithm for these two cases takes a significant amount of computation time because of its recursive nature. Consequently, they also develop heuristic methods to find approximate solutions for these two objective functions using earliest due date (EDD) and smallest slack time (SST) rules to sequence the jobs. Computational investigation is

presented to show that their heuristic performs better than the EDD rule for the maximum tardiness problem, whereas it does not perform as well for minimizing the number of tardy jobs.

2.2.3 Job Shop

In a job shop, each job has its own route or processing over the machines. Dauzere-Peres and Lasserre [20] first addressed lot streaming in a more general job shop environment. They presented an iterative procedure that simultaneously computes sublot sizes and the sequence of sublots on the machines. Using linear programming that minimizes the makespan, they determined the optimal sublot sizes given a fixed number of sublots and sequence of sublots on the machine for the lot streaming problem. In the job shop scheduling problem, they determined the sequence of the sublots on the machines by minimizing makespan given fixed sizes of sublots. They investigate the effect the fixing of number of sublots has on the performance of the lot streaming procedure. The procedure they present to solve the lot streaming problem for the general job shop case considers only makespan; however, with additional research, they state their methodology could be applied to other objective criteria.

Buscher and Shen [12] have extended the use of lot streaming to a job shop environment through the use of an integrated tabu search algorithm. They consider the $m/n/\{C,E\}/DV/C_{max}$ lot streaming job shop problem. They represent the problem as a disjunctive graph. The critical path of the graph gives the makespan. A three-phase method is used to find near optimal solutions to the problem. Sublot sizes are determined in Phase 1. Then, in Phase 2, a schedule is determined by using a tabu search. Sublot sizes are then varied to obtain improvements on the makespan value by using tabu search. Computational results are presented, which show that their method generates near-optimal solution efficiently.

Edis and Ornek [31] have applied lot streaming to the following problem:

$m/n/E/II/DV/\{C_{max}, \sum C_j, \sum U_j\}/Setup$ job shop problem, where U_j is the number of tardy jobs. They also consider minimizing the flow time of sublots and the number of tardy sublots. This lot streaming problem is different in the sense that stochastic components are considered. They use simulation to determine the effects lot streaming and different transportation queue disciplines would have on the aforementioned performance measures. A search-based heuristic is proposed to determine sublot allocations.

Huang [39] approaches a multi-objective job shop problem using lot streaming. The objective functions considered are minimization of weighted total stock, machine idle cost, and carrying costs. Using an ant colony optimization algorithm to solve the lot streaming application to this problem, the solutions obtained are shown to be consistent with the optimal solutions obtained using a linear program. The heuristic is also shown to be faster than the use of LINGO.

Buscher and Chen [13] have addressed a lot streaming problem in a job shop environment with setups, while minimizing makespan. They present an integer programming formulation of the problem with a few extensions. The problem is solved directly using an optimization software, thus only small instances of their formulation are able to be solved in a reasonable amount of time. Computational results are presented to depict the advantage of using lot streaming in a job shop environment.

Liu [52] has explored the benefits of lot streaming in a job shop problem with job values exponentially deteriorating overtime. A deteriorating job is defined to require more time to process if the job is not started sooner. The problem is approached in two parts. First, lot streaming is used to determine the subjob combinations, and secondly, all the subjobs are scheduled in the job shop environment. The optimal number of subjobs of a single job is determined using computational simulation. A genetic algorithm is used to, first

determine all the subjob combinations, and then, schedule, all the subjobs of the job shop using simple dispatching rules. The proposed approach can additionally handle non-equal sized subjobs.

2.2.4 Open Shop

Lot streaming has also been applied in the open shop manufacturing environment. Hall *et al.* [38] discuss the two-machine open shop lot streaming problem to minimize makespan with no-wait and n products. In the open shop manufacturing environment, the jobs can be performed on either of the two machines in any order. They show that selecting a product schedule with integer-sized sublots can be modeled by using a generalized traveling salesman problem (GTSP). Thus, they develop a heuristic using the GTSP to minimize the makespan, which can efficiently solve lot streaming problems up to 50 products [38].

2.2.5 Assembly

Recently, the use of lot streaming has been extended to the assembly manufacturing environment. Dastidar and Nagi [19] approach a batch splitting problem in an assembly scheduling environment. In an assembly environment, there is an end item that is made up of various parts that have to be manufactured before the end item can be assembled. Thus, there are precedence constraints that are represented using a network. Through a simple example, they show that lot splitting could have beneficial effects on minimizing the makespan by streaming operations. Two heuristics are developed where one splits the lots so that the makespan is minimized on parallel machines with preemptions, while the second heuristic uses the lot splitting heuristic and determines the schedule. The schedule the second heuristic determines is based on the critical path. They also report that their two heuristics generate efficient solutions as opposed to the model formulation of the problem solved using CPLEX

7.1. heuristics.

Chan, Wong and Chan [16] address a product assembly problem using lot streaming in a job shop environment. A GA is used to split the lots optimally over all possible combinations followed by simple dispatching rules (SDRs) to solve the assembly job shop scheduling problem (AJSP). The AJSP consists of three stations where one station contains m distinct machines, and there are p product types for each planning period. The type of lot streaming problem they consider is $m/n/\{C, E\}/II/DV/Z$ where Z represents the tardiness and inventory costs. After using the GA to produce all possible sublot combinations, the SDRs proposed to solve the AJSP are shortest processing time (SPT), longest processing time (LPT), minimum slack time (MST) and earliest due date (EDD). The MST rule with equal-sized sublots is shown to give the best performance with respect to cost objectives when compared with other rules for both equal and consistent sublots.

Wong, Chan and Chan [75] extend their work in [74] to a resource-constrained assembly job shop scheduling problem. A GA approach is used to solve the problem, though they also present comparative results for GA and Particle Swarm Optimization (PSO) methods. The GA approach is shown to outperform the PSO benchmark for equal sublots on every test problem with respect to computational effort and the performance measure.

Chapter 3

Formulation of a Mathematical Model for the AFGP

Since the officer and enlisted education systems are completely separate, they are modeled as two systems, though with very similar in logic. As explained in the Introduction, officers have one training requirement before they are eligible to be assigned to a unit. This training requirement, BOLC B, is specific to each officer's AOC. For some AOC, multiple concurrent sessions may be permitted depending on resource capacity. The model formulation views the aggregate ability to offer classes concurrently as the number of instructors. More explicitly, if there are enough resources to offer two classes concurrently regardless the total requirements needed, the model formulation views this capacity as two available instructors.

The enlisted soldiers have a very similar process as they obtain more training for their specific MOS, and then, assigned to a unit, as previously detailed in Chapter 1. In addition to being trained specific to their MOS, enlisted soldiers first enter the system as civilian volunteers, and thus, must begin the soldierization process with basic training. Since BT is a requirement for all enlisted soldiers no matter their skill set, soldiers of various MOS are taught in the same training sessions. We have monthly recruit data for each skill set and assume a uniform distribution to determine the daily arrival of people entering the system. Overall, there is sufficient BT resources for all soldiers. Therefore, there is no constraint restricting a classroom maximum or minimum size requirement for BT classes. Although

there are sufficient BT resources to accommodate soldiers entering the system, there are not sufficient AIT resources to accommodate all the soldiers completing BT. With the seasonality of recruit surges, some enlisted soldiers may be required to wait before entering BT to account for the constrained resources of AIT respective of MOS. As the enlisted volunteers enter the system, a class is formed with a start time such that the wait time between entering the system and starting the class for every soldier is less than or equal to a fixed value, ED. The maximum time a new volunteer can be delayed (ED) from starting BT with minimum negative effect on the quality, is being determined by ongoing research.

As the BT classes end, the class of enlisted soldiers separate to attend the remaining of their training specific to their MOS or AIT. Similar to the officer BOLC B setup, each skill set has a given capacity of resources that allows classes to be offered simultaneously, just like parallel machines. The model formulation treats the capacity to offer more than one class concurrently as the number of instructors. The number of classes that can be offered concurrently varies across MOS. The training duration for each class of a given MOS is the same; however, training duration across MOS varies depending on the technical sophistication involved. In order to ensure the desired quality of training, there are minimum and maximum class capacity ratios set by TRADOC.

The third stage of the model has no processing time and is more or less the departure of both officers and enlisted soldiers from the IMT system after training is completed. Soldiers depart the system when they are assigned to a unit in stage three. There are two types of demands the Army must satisfy with the newly graduated recruits: units following ARFORGEN (as explained in Chapter 1) and all the other demands across the Army. The demands across the rest of the Army not following ARFORGEN have time constraints depending on the position. We address this by analyzing the average monthly demand for all positions the Army requires. For the units following ARFORGEN, when units enter the Reset/Train phase, there is only a short period of time to re-form the unit. The model gives a priority to the ARFORGEN requirements by forcing the exact number of positions re-

quired for the different skill sets for both officer and enlisted soldier to be fulfilled during the given time period. This unit assignment stage accounts for the time windows the units have available to fulfill their manning requirements for each AOC or MOS for both officer and enlisted soldier, respectively, according to the MTOE for both the ARFORGEN units and the remaining Army requirements. The model allows for some flexibility when meeting the requirements not following a strict ARFORGEN schedule. Similar to an inventory model, the non-ARFORGEN requirements are allowed to be fulfilled later or soldiers are allowed to wait until the positions requiring their skill sets become open. A soldier can only depart the training system when he or she is assigned to a unit. For both the ARFORGEN units and non-ARFORGEN units, if a soldier's completion time is before the beginning window of the unit he or she is assigned to, the soldier incurs a wait time, thus increasing the soldier's total time in system.

The initial military training system can be addressed by lot streaming as a three-stage flexible flow shop problem with m identical machines for each of the n skill sets in Stage 2, where the Advanced Individual Training occurs for enlisted soldiers and the Basic Officer Leadership Course (BOLC) B occurs for officers. Each MOS within the enlisted training system and AOC for the officer education system are considered as the lots and the classes are considered as sublots.

3.1 Model Formulation

The notation for AFGP is extensive. We provide guidelines to read the notation as follows:

- The same letter is used to depict parameters and variables for soldiers and officers except that the one related to officers is prefixed by "O." For example, letter P stands for processing time of a soldier while OP stands for processing of an officer. Likewise, σ is the skill set for soldiers, while $O\sigma$ is the skill set for officers.

- The superscript of a letter depicts the stage. For example, $P^{(1)}$ belongs to Stage 1 while $P^{(2)}$ belongs to Stage 2.

- The subscript of a letter depicts its association with a solider (s) or officer (o), skill set (l), BCT window (b), class (k), or instructor (a).

Notation.

Parameters

T	Total number of time periods (months) included in the model
S	Total number of enlisted soldiers
σ	Total number of skill sets for enlisted soldier
R_s	Ready time of an enlisted soldier s for Basic Training, $\forall\ s = 1, ..., S$
D	Maximum number of time periods an enlisted soldier can wait to start the first class (Basic Training)
$P_s^{(1)}$	Processing time of a Basic Training class for soldier s, $\forall\ s = 1, ..., S$
$B^{(1)}$	Upper bound on the number of classes for Basic Training $= T/D$
$I^{(2)}$	Total number of instructors for Advanced Individual Training (AIT) of the enlisted training system
$P_l^{(2)}$	Processing time of an AIT class for skill set l, $\forall\ l = 1, ..., \sigma$
$\gamma_{la}^{(2)}$	Penalty cost used to choose an earliest available instructor as an AIT instructor a of skill $l, \forall\ a = 1, ..I_l^{(2)},\ l = 1, ..., \sigma$
$L^{(2)}$	Maximum sublot size allowed in AIT based on instructor-student ratio
$U^{(2)}$	Minimum sublot size allowed in AIT based on instructor-student ratio
$B^{(2)}$	Upper bound on the number of classes for AIT ($= \lfloor S/L^{(2)} \rfloor$, where (x) is the largest integer smaller than x)
ρ	Penalty cost for non-ARFORGEN assignments when soldiers are early or tardy
β	Total number of ARFORGEN BCTs in Unit Assignment
B	Total number of BCTs and monthly Army demand Unit Assignments
Req_{lb}	Enlisted personnel of skill l required in BCT Army Unit b, $\forall\ l = 1, ..., \sigma$, $b = 1, ..., B$
OS	Total number of officers
$O\sigma$	Total number of skill sets for officers
OR_o	Ready time of officer o for BOLC, $\forall\ o = 1, ..., OS$
$OI_l^{(1)}$	Total number of instructors for Basic Officer Leadership Course (BOLC) l of the officer training system, $\forall\ l = 1, ..., \sigma$
$OP_l^{(1)}$	The processing time of BOLC, $l, \forall\ l = 1, ..., \sigma$
$OU_l^{(1)}$	Maximum sublot size allowed for BOLC l based on instructor-student ratio $\forall\ l = 1, ..., \sigma$
$OL_l^{(1)}$	Minimum sublot size allowed for BOLC l based on instructor-student ratio $\forall\ l = 1, ..., \sigma$

$OB_l^{(1)}$	Upper bound on the number of classes for BOLC l $(= \lfloor OS/OL^{(1)} \rfloor)$, $\forall\ l = 1, ..., \sigma$
$OReq_{lb}$	Officer personnel of skill l required in BCT b during Unit Assignment $\forall\ l = 1, ..., \sigma,\ b = 1, ..., B$
BW_b	Beginning window of BCT b, $\forall\ b = 1, ..., B$
EW_b	Ending window of BCT b, $\forall\ b = 1, ..., B$
G	A large positive number

Variables

Enlisted Soldier Training System

$W_s^{(1)}$	The time an enlisted soldier s waits to start BT, $\forall\ s = 1, ..., S$
$W_s^{(2)}$	The time an enlisted soldier s waits to start AIT, $\forall\ s = 1, ..., S$
$W_s^{(3)}$	The time an enlisted soldier s waits to be assigned to a unit, $\forall\ s = 1, ..., S$
$tW_s^{(3)}$	The difference in time from a soldier's completion of AIT and the beginning window of the unit the solider is assigned, $\forall\ s = 1, ..., S$
$S_i^{(1)}$	Size of Basic Training class i, $\forall\ i = 1, ..., B^{(1)}$
$S_{lka}^{(2)}$	Size of class k for skill l with instructor a (of AIT), $\forall\ l = 1, ..., \sigma,\ k = 1, ..., B^{(2)},\ a = 1, ..., I^{(2)}$
$S_{lb}^{(3)}$	Number of enlisted soldiers of skill l assigned to BCT b, $\forall\ l = 1. ..., \sigma,\ b = 1, ..., B$
α_{lb}^{+}	Number of enlisted soldiers of skill l in excess of requirements for Army units, $b = \beta + 1, ...B\ \forall\ l = 1, ..., \sigma$
α_{lb}^{-}	Number of enlisted soldiers of skill l not meeting requirements for Army units, $b = \beta + 1, ...B\ \forall\ l = 1, ..., \sigma$
$C_i^{(1)}$	Completion time of Basic Training class i, $\forall\ i = 1, ..., B^{(1)}$
$C_{lka}^{(2)}$	Completion time of AIT class k of skill set l with instructor a, $\forall\ l = 1, ..., \sigma,\ k = 1, ..., B^{(2)},\ a = 1, ..., I^{(2)}$

$$Y_{slika}^{(2)} = \begin{cases} 1, & \text{if enlisted soldier } s \text{ of skill } l \text{ is in both AIT class } k \text{ with instructor } a \text{ and in BT class } i \\ 0, & \text{otherwise, } \forall\ s = 1, ..., S,\ l = 1, ..., \sigma,\ i = 1, ..., B^{(1)}, \\ & k = 1, ..., B^{(2)},\ a = 1, ..., I^{(2)}. \end{cases}$$

$$Y_{slkab}^{(3)} = \begin{cases} 1, & \text{if enlisted soldier } s \text{ of skill } l \text{ is in both AIT class } k \text{ with instructor } a \text{ and BCT } b \\ 0, & \text{otherwise, } \forall\ s = 1, ..., S,\ l = 1, ..., \sigma,\ k = 1, ..., B^{(2)}, \\ & a = 1, ..., I^{(2)},\ b = 1, ..., B. \end{cases}$$

$$X_{lka}^{(2)} = \begin{cases} 1, & \text{if an AIT skill } l \text{ class } k \text{ is formed with instructor } a \\ 0, & \text{otherwise, } \forall\ l = 1, ..., \sigma,\ k = 1, ..., B^{(2)},\ a = 1, ..., I^{(2)}. \end{cases}$$

$$Z_{sli}^{(1)} = \begin{cases} 1, & \text{if enlisted soldier } s \text{ of skill set } l \text{ is in BT class session } i \\ 0, & \text{otherwise, } \forall\ s = 1, ..., S,\ l = 1, ..., \sigma,\ i = 1, ..., B^{(1)}. \end{cases}$$

$$Z_{slka}^{(2)} = \begin{cases} 1, & \text{if enlisted soldier } s \text{ with skill set } l \text{ is in AIT class session } k \\ & \text{with instructor } a \\ 0, & \text{otherwise, } \forall\ s = 1, ..., S, l = 1, ..., \sigma,\ \ k = 1, ..., B^{(2)}, \\ & a = 1, ..., I^{(2)}. \end{cases}$$

$$Z_{slb}^{(3)} = \begin{cases} 1, & \text{if enlisted soldier } s \text{ of skill } l \text{ is assigned to BCT } b \\ 0, & \text{otherwise, } \forall\ s = 1, ..., S,\ l = 1, ..., \sigma,\ \ b = 1, ..., B. \end{cases}$$

For the $X_{lka}^{(2)}$ indicator variable, note that

$$X_{lka}^{(2)} = \begin{cases} 1, & \text{if } S_{lka}^{(2)} > 0 \\ 0, & \text{if } S_{lka}^{(2)} = 0. \end{cases}$$

Officer Training System

$OW_o^{(1)}$	Time an officer waits to start BOLC, $\forall\ o = 1, ..., OS$
$OW_o^{(3)}$	Time an officer waits to be assigned to a unit, $\forall\ o = 1, ..., OS$
$OS_{lej}^{(1)}$	Size of class e for skill l with instructor j of BOLC, $\forall\ l = 1, ..., O\sigma,\ e = 1, ..., OB_l^{(1)},\ j = 1, ..., OI_l^{(1)}$
$OS_{lb}^{(3)}$	Number of officers of skill l assigned to BCT b, $\forall\ l = 1, ..., O\sigma$, $b = 1, ..., B$
$OC_{lej}^{(1)}$	Completion time of BOLC class e for skill set l with instructor j, $\forall\ e = 1, ..., OB_l^{(1)},\ l = 1, ..., O\sigma,\ j = 1, ..., OI_l^{(1)}$
$O\alpha_{lb}^{+}$	Number of officers of skill l in excess of the $b = \beta + 1, ...B$, requirements for Army unit b, $\forall\ l = 1, ..., \sigma$
$O\alpha_{lb}^{-}$	Number of officers of skill l not meeting the requirements for Army units, $b = \beta + 1, ...B$, $\forall\ l = 1, ..., \sigma$

$$OY_{olejb}^{(3)} = \begin{cases} 1, & \text{if officer } o \text{ of skill } l \text{ is in BOLC class } e \text{ with instructor } j \text{ and BCT } b \\ 0, & \text{otherwise, } \forall\ o = 1, ..., OS,\ l = 1, ..., O\sigma, \\ & e = 1, ..., OB_l^{(1)},\ j = 1, ..., OI_l^{(1)}\ b = 1, ..., B. \end{cases}$$

$$OX_{lej} = \begin{cases} 1, & \text{when BOLC class } e \text{ of skill set } l \text{ is formed with instructor } j \\ 0, & \text{otherwise, } \forall\ l = 1, ..., O\sigma,\ e = 1, ..., OB_l^{(1)},\ j = 1, ..., OI_l^{(1)}. \end{cases}$$

$$OZ^{(1)}_{olej} = \begin{cases} 1, & \text{if officer } o \text{ of skill } l \text{ is in BOLC class } e \text{ with instructor } j \\ 0, & \text{otherwise, } \forall\ o = 1, ..., OS,\ l = 1, ..., O\sigma, \\ & e = 1, ..., OB^{(1)}_l,\ j = 1, ..., OI^{(1)}_l \\ . & \end{cases}$$

For the OX_{lej} indicator variable, note

$$OX_{lej} = \begin{cases} 1, & \text{if } OS^{(1)}_{lej} > 0 \\ 0, & \text{if } OS^{(1)}_{lej} = 0. \end{cases}$$

$$OZ^{(3)}_{olb} = \begin{cases} 1, & \text{if officer } o \text{ with skill set } l \text{ is assigned to unit } b \text{ of BCT } b \\ 0, & \text{otherwise, } \forall\ o = 1, ..., OS,\ l = 1, ..., O\sigma,\ b = 1, ..., B. \end{cases}$$

Next, we present a model for the Army Force Generation Problem (AFGP).

Model AFGP

Minimize:

$$\begin{aligned} FT = & \sum_{l=1}^{\sigma} \sum_{s \in s_l} (W^{(1)}_s + P^{(1)}_s) + \sum_{l=1}^{\sigma} \sum_{s \in s_l} (W^{(2)}_s + P^{(2)}_l) + \sum_{s=1}^{S} W^{(3)}_s \\ & + (\rho) \left(\sum_{l=1}^{\sigma} \sum_{b=\beta+1}^{B} (\alpha^{+}_{lb} + \alpha^{-}_{lb}) + \sum_{l=1}^{O\sigma} \sum_{s \in s_l} (O\alpha^{+}_{lb} + O\alpha^{-}_{lb}) \right) \\ & + \sum_{l=1}^{\sigma} \sum_{a=1}^{I^{(2)}} \sum_{k=1}^{B^{(2)}} \gamma_{la} X^{(2)}_{lka} + \sum_{o=1}^{OS} OW^{(1)}_o + OP^{(1)} + \sum_{o=1}^{OS} OW^{(3)}_o \end{aligned}$$

The first three and last last three terms capture the time in the system for enlisted soldiers and officers, respectively. The fourth term accounts for the excesses or shortages of Army soldiers (both enlisted and officers) from the required amount, while the fifth term ensures the use of the earliest available AIT instructors.

Enlisted Soldier Training

Basic Training

During the first stage, civilian volunteers enter the system to begin the soldierization

process with basic training.

Using the indicator variable, $Z_{sli}^{(1)}$, the enlisted soldier is restricted to be in exactly one BT class:

$$\begin{aligned} \sum_{i=1}^{B^{(1)}} Z_{sli}^{(1)} = 1, \\ \forall\ s = 1, ..., S,\ l = 1, ..., \sigma. \end{aligned} \tag{3.1}$$

The time the soldiers wait to start a BT class is determined using the following constraint:

$$\begin{aligned} 0 \leq C_i^{(1)} - P^{(1)} - R - G(1 - Z_{sli}^{(1)}) \leq W_s^{(1)}, \\ \forall\ s = 1, ..., S,\ l = 1, ..., \sigma,\ i = 1, ..., B^{(1)}. \end{aligned} \tag{3.2}$$

The wait time of the soldiers entering the BT class has to be less than the parameter ED which is the maximum delay a soldier can have before starting BT after entering the system:

$$\begin{aligned} 0 \leq W_s^{(1)} \leq ED, \\ \forall\ s = 1, ..., S. \end{aligned} \tag{3.3}$$

In order to make sure a previous class is finished before the next class begins, the following constraints are needed.

$$C_1^{(1)} - P_s^{(1)} \geq 0, \tag{3.4}$$

$$\begin{aligned} C_{i+1}^{(1)} - C_i^{(1)} - P_s^{(1)} \geq 0, \\ \forall\ i = 1, ..., B^{(1)}. \end{aligned} \tag{3.5}$$

Soldiers cannot attend a BT class whose start time is earlier than the arrival time of the respective soldier. Since each soldier can have a different availability time, the following two constraints account for assigning a soldier to a class only if its start time is greater than the soldier's availability time. If the soldier is not assigned to the particular class, then a constraint is relaxed.

a. First BT Class

$$C_1^{(1)} - P_s^{(1)} - R_s + G(1 - Z_{sl1}^{(1)}) \geq 0, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma; \tag{3.6}$$

b. For BT classes 2,...,$B^{(1)}$.

$$C_i^{(1)} - P_s^{(1)} - R_s + G(1 - Z_{sli}^{(1)}) \geq 0, \quad \forall\ i = 2, ..., B^{(1)} - 1,\ s \in s_l,\ l = 1, ..., \sigma, \tag{3.7}$$

where s_l is the set of soldiers requiring skill set l. Note that no constraint is required for the completion time of the last class, $B^{(1)}$

Advanced Individual Training

Unlike BT, the AIT stage requires the offering of classes and use of resources respective of a skill set. For each skill set, the training resources are completely separate. Each soldier requires training in exactly one skill set. The restriction that one enlisted soldier can only be in one class of an instructor of the requisite AIT skill set course is as follows:

$$\sum_{k=1}^{B_l^{(2)}} \sum_{a=1}^{I_l^{(2)}} Z_{slka}^{(2)} = 1, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{3.8}$$

The sum of enlisted officers of skill set l in a class k per instructor equals the class size for instructor a:

$$\sum_{s \in s_l} Z^{(2)}_{slka} = S^{(2)}_{lka}, \quad \forall\ l = 1, ..., \sigma,\ k = 1, ..., B^{(2)}_l, a = 1, ..., I^{(2)}_l. \tag{3.9}$$

A class k can be formed with any instructor a. To account for the possibility of formation of class k of skill set l by an instructor a, the indicator variable $X^{(2)}_{lka}$ can be at most one over the sum of all instructors.

$$\sum_{a=1}^{I^{(2)}_l} X^{(2)}_{lka} \leq 1, \quad \forall\ l = 1, ..., \sigma,\ k = 1, ..., B^{(2)}_l. \tag{3.10}$$

Given the different combinations of class and instructors that can be formed, a natural precedence to enforce is that the classes that did form (represented by its nonzero size) should precede the classes that did not form. To account for this precedence, the binary $X^{(2)}_{lka}$ variable is used in the following constraint.

$$\sum_{a=1}^{I^{(2)}_l} X^{(2)}_{lka} \leq \sum_{a=1}^{I^{(2)}_l} X^{(2)}_{l(k-1)a}, \quad \forall\ l = 1, ..., \sigma,\ k = 2, ..., B^{(2)}_l. \tag{3.11}$$

Unlike BT, the classes for the different skill sets in AIT have minimum and maximum class size requirements. With the class formation indicator variable $X^{(2)}_{lka}$, the size restrictions are set as shown in the following constraints.

$$L^{(2)}(X^{(2)}_{lka}) \leq S^{(2)}_{lka}, \quad \forall\ l = 1, ..., \sigma,\ k = 1, ..., B^{(2)}_l\ a = 1, ..., I^{(2)}_l. \tag{3.12}$$

$$U^{(2)}(X^{(2)}_{lka}) \geq S^{(2)}_{lka}, \quad \forall\ l = 1, ..., \sigma,\ k = 1, ..., B^{(2)}_l\ a = 1, ..., I^{(2)}_l. \tag{3.13}$$

The sum of class sizes over the the total number of AIT instructors must be equal to the total number of enlisted soldiers:

$$\sum_{k=1}^{B^{(2)}_l} \sum_{a=1}^{I^{(2)}_l} S^{(2)}_{lka} = S, \quad \forall\ l = 1, ..., \sigma. \tag{3.14}$$

As soldiers finish BT at varying times, they may incur a wait time before starting the AIT class respective of their skill set. The time a soldier has to wait between finishing BT and starting an AIT class is determined by the following constraint. The variable $Y^{(2)}_{slika}$ indicates when a soldier s with skill set l is in both the BT class i and the AIT class k with instructor a.

$$0 \leq C^{(2)}_{lka} - P^{(2)}_l - C^{(1)}_i - G(1 - Y^{(2)}_{slika}) \leq W^{(2)}_s, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., B^{(1)}, \quad k = 1, ..., B^{(2)}_l,\ a = 1, ..., I^{(2)}_l. \tag{3.15}$$

The following constraints ensure that a class cannot use the same instructor until the previous class is finished. The first constraint serves as initialization.

$$C^{(2)}_{l11} - C^{(2)}_{l01} - P^{(2)}_l + G(1 - X^{(2)}_{l11}) \geq 0, \quad \forall\ l = 1, ..., \sigma. \tag{3.16}$$

$$C^{(2)}_{lka} - C^{(2)}_{lk'a} - P^{(2)}_{l} + G(1 - X^{(2)}_{lka}) \geq 0,$$
$$\forall\ l = 1, ..., \sigma,\ k = 2, ..., B^{(2)}_{l},\ k' = 1, ..., B^{(2)}_{l},\ k' < k, \tag{3.17}$$
$$a = 1, ..., I^{(2)}_{l}.$$

The following constraint is used to ensure that the classes are numbered with respect to their completion times such that the last class has the latest completion time.

$$C^{(2)}_{lka} - C^{(2)}_{lk'a'} + G(1 - X^{(2)}_{lka}) \geq 0,$$
$$\forall\ l = 1, ..., \sigma,\ k = 2, ..., B^{(2)}_{l},\ k' = 1, ..., B^{(2)}_{l},\ k' < k, \tag{3.18}$$
$$a = 1, ..., I^{(2)}_{l}, a' = 1, ...I^{(2)}_{l}.$$

The following constraint assures the classes of size zero have zero completion times:

$$C^{(2)}_{lka} \leq G(X^{(2)}_{lka}),$$
$$\forall\ k = 1, ..., B^{(2)}_{l},\ a = 1, ..., I^{(2)}_{l},\ l = 1, ..., \sigma. \tag{3.19}$$

With the current set of constraints, the model can generate alternative optimal schedules of assigning classes to instructors. These alternative optimal schedules can be avoided through the use of symmetry breaking constraints by numbering the instructors and picking them in the increasing order as follows:

$$\sum_{k=1}^{B^{(2)}_{l}} X^{(2)}_{lk(a-1)} \geq \sum_{k=1}^{B^{(2)}_{l}} X^{(2)}_{lka},$$
$$\forall\ a = 2, ..., I^{(2)}_{l},\ l = 1, ..., \sigma. \tag{3.20}$$

Note that the binary variable $Y^{(2)}_{slika}$, which is a multiplication of variables $Z^{(2)}_{slka}$ and $Z^{(1)}_{sli}$, can be linearized by using the following inequalities:

$$Z^{(2)}_{slka} \geq \sum_{i=1}^{B^{(1)}} Y^{(2)}_{slika}, \quad \forall\ s \in s_l, \quad a = 1, ..., I^{(2)}_l,\ k = 1, ..., B^{(2)}_l,\ l = 1, ..., \sigma. \tag{3.21}$$

$$Z^{(1)}_{sli} \geq \sum_{k=1}^{B^{(2)}_l} \sum_{a=1}^{I^{(2)}_l} Y^{(2)}_{slika}, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., B^{(1)}. \tag{3.22}$$

$$Z^{(1)}_{sli} + Z^{(2)}_{slka} - 1 \leq Y^{(2)}_{slika}, \quad \forall\ s \in s_l,\ i = 1, ..., B^{(1)}, \quad a = 1, ..., I^{(2)}_l,\ k = 1, ..., B^{(2)}_l,\ l = 1, ..., \sigma, . \tag{3.23}$$

The following constraints allow the classes of variable sizes to be formed under the restriction that a soldier cannot be assigned to an AIT class before completing his BT class.

a. First Class of AIT

$$C^{(2)}_{l1a} - P^{(2)}_l - C^{(1)}_i + G(1 - Y^{(2)}_{sli1a}) \geq 0, \quad \forall\ s \in s_l,\ i = 1, ..., B^{(1)},\ a = 1, ..., I^{(2)}_l,\ l = 1, .., \sigma \tag{3.24}$$

b. For AIT classes $2, ..., B^{(2)}_l$

$$C^{(2)}_{lka} - P^{(2)}_l - C^{(1)}_i + G(1 - Y^{(2)}_{slika}) \geq 0, \quad \forall\ s \in s_l,\ i = 1, ..., B^{(1)},\ a = 1, ..., I^{(2)}_l,\ k = 1, ..., B^{(2)}_l,\ l = 1, ..., \sigma.. \tag{3.25}$$

Unit Assignment

After the enlisted soldiers complete training in both BT and AIT, they are assigned to an available unit. There is no processing that takes place during this stage; however, the requisite number of soldiers of various skill sets must be allocated for each unit within its

specified time window. A binary variable is used to keep track of which enlisted soldier is assigned to which unit. An enlisted soldier can be assigned to only one unit:

$$\sum_{b=1}^{B} Z_{slb}^{(3)} = 1, \quad \forall\ s \in s_l, l = 1, ..., \sigma. \tag{3.26}$$

The number of enlisted soldiers assigned to an ARFORGEN BCT must meet the manning requirement:

$$\sum_{s \in s_l} Z_{slb}^{(3)} = Req_{lb}, \quad \forall\ l = 1, ..., \sigma,\ b = 1...\beta. \tag{3.27}$$

Besides ARFORGEN BCT units, the Army must also fulfil demand requirements for both the Generating Force and Operating Force. Unlike the ARFORGEN demand, the time requirements of fulfilling the rest of the Army demand are not as severe. Therefore, the following two constraints allow for shortages of soldiers for some units only to be fulfilled later with an associated penalty cost in the objective function. The constraints also allow for additional soldiers to carry over to future unit demands. The first of the two constraints listed below sets up the initial excess of soldiers as zero and the second constraint sets up how the α^+ and α^- values track the excesses and shortages of soldiers graduating IMT.

$$\alpha_{l(\beta+1)}^{+} - \alpha_{l(\beta+1)}^{-} = \sum_{s \in s_l} Z_{sl(\beta+1)}^{(3)} - Req_{l(\beta+1)}, \quad \forall\ l = 1, ..., \sigma. \tag{3.28}$$

$$\alpha_{lb}^{+} - \alpha_{lb}^{-} = \alpha_{l(b-1)}^{+} - \alpha_{l(b-1)}^{-} + \sum_{s \in s_l} Z_{slb}^{(3)} - Req_{lb}, \quad \forall\ l = 1, ..., \sigma,\ b = \beta + 2...B. \tag{3.29}$$

The symmetry breaking constraint for the third stage, or the unit assignment stage, ensures that a soldier is assigned to the first available unit in order to reduce the other symmetric optimal solutions.

$$\sum_{b=1}^{B} BW_b(Z^{(3)})_{slb} \leq \sum_{b=1}^{B} BW_b(Z^{(3)}_{(s+1)lb}), \quad \forall\ s = 1, ..., |s_l| - 1, l = 1, ..., \sigma. \tag{3.30}$$

The binary variable $Y^{(3)}_{slkab}$, which is a product of $Z^{(2)}slka$ and $Z^{(3)}_{slk}$ can be linearized using the following in equations:

$$Z^{(2)}_{slka} \geq \sum_{b=1}^{B} Y^{(3)}_{slkab}, \quad \forall\ s \in s_l,\ k = 1, ..., B^{(2)},\ a = 1, ..., I^{(2)},\ b = 1, ..., B. \tag{3.31}$$

$$Z^{(3)}_{slb} \geq \sum_{k=1}^{B^{(2)}} \sum_{a=1}^{I^{(2)}} Y^{(3)}_{slkab}, \quad \forall\ s \in s_l,\ k = 1, ..., B^{(2)}, \quad a = 1, ..., I^{(2)},\ b = 1, ..., B,\ l = 1, ..., \sigma. \tag{3.32}$$

$$Z^{(2)}_{slka} + Z^{(3)}_{slb} - 1 \leq Y^{(3)}_{slkab}, \quad \forall\ s \in s_l,\ k = 1, ..., B^{(2)},\ a = 1, ..., I^{(2)},\ b = 1, ..., B,\ l = 1, ..., \sigma. \tag{3.33}$$

The time a soldier waits before being assigned to a unit is determined by taking the difference between the beginning of the window the unit is assigned to and the completion time of the AIT class he/she came from. Thus, this value will be negative if the soldier finished the AIT class earlier than the beginning of the window. We determine a soldier's waiting time, $W^{(3)}_s$ as follows:

$$BW_b - C^{(2)}_{lka} - G(1 - Y^{(3)}_{slkab}) \leq W^{(3)}_s, \\ \forall\ s \in s_l,\ l = 1, ..., \sigma,\ b = 1, .., B, \\ k = 1, ..., B^{(2)},\ a = 1, ..., I^{(2)}. \tag{3.34}$$

$$W^{(3)}_s \geq 0 \\ \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{3.35}$$

The soldiers meeting the manning requirement of the BCT must also complete AIT within the window the BCT is given to man units:

$$EW_b - C^{(2)}_{lka} + G(1 - Y^{(3)}_{slkab}) \geq 0, \\ \forall\ s \in s_l,\ l = 1, ..., \sigma,\ k = 1, ..., B^{(2)}, \\ a = 1, ..., I^{(2)},\ b = 1, ..., \beta. \tag{3.36}$$

Officer Soldier Training

Basic Officer Leadership Course B

The restriction that one officer can only be in one class for each BOLC B course is as follows:

$$\sum_{e=1}^{OB^{(1)}_l} \sum_{j=1}^{OI^{(1)}_l} OZ^{(1)}_{olej} = 1, \\ \forall\ o \in o_l\ l = 1, ..., O\sigma, \tag{3.37}$$

where o_l is the set of officers with skill set l. The sum of officers in a class per instructor equals the class size for instructor j:

$$\sum_{o \in s_l} OZ^{(1)}_{olej} = OS^{(1)}_{lej}, \quad \forall\ l = 1, ..., O\sigma,\ e = 1, ..., OB^{(1)}, j = 1, ..., OI^{(1)}. \tag{3.38}$$

Each class, e, for skill set, l can be offered by only one instructor of that skill set.

$$\sum_{j=1}^{OI^{(1)}_l} OX^{(1)}_{lej} \leq 1, \quad \forall\ e = 1, ..., OB^{(1)}_l,\ l = 1, ..., \sigma. \tag{3.39}$$

Given the different combinations of class and instructors that can form, a natural precedence to enforce is that the classes that did form (represented by its nonzero size) should precede the classes that did not form. To account for this precedence, the binary indicator variable $OX^{(1)}_{lej}$ is used as shown in the following constraint.

$$\sum_{j=1}^{OI^{(1)}_l} OX^{(1)}_{lej} \leq \sum_{j=1}^{OI^{(1)}_l} OX^{(1)}_{l(e-1)j}, \quad \forall\ k = 2, ..., B^{(2)}_l,\ l = 1, ..., O\sigma. \tag{3.40}$$

The class size must be within the minimum and maximum classroom size restrictions:

$$OL^{(1)}_l(OX_{lej}) \leq OS^{(1)}_{lej}, \quad \forall\ l = 1, ..., O\sigma,\ e = 1, ..., OB^{(1)}\ j = 1, ..., OI^{(1)}. \tag{3.41}$$

$$OU^{(1)}_l(OX_{lej}) \geq OS^{(1)}_{lej}, \quad \forall\ l = 1, ..., O\sigma,\ e = 1, ..., OB^{(1)}\ j = 1, ..., OI^{(1)}. \tag{3.42}$$

The sum of the classes over the total number of BOLC B instructors must be equal to the total number of officers:

$$\sum_{e=1}^{OB_l^{(1)}} \sum_{j=1}^{OI_l^{(1)}} OS_{lej}^{(1)} = OS, \quad \forall\, l = 1, ..., O\sigma. \tag{3.43}$$

The time an officer waits before starting BOLC B is determined using the following:

$$0 \leq OC_{lej}^{(1)} - OP_l^{(1)} - OR_o - G(1 - OZ_{olej}^{(1)}) \leq OW_o^{(1)} \quad \forall\, o \in o_l,\ l = 1, ..., O\sigma,\ e = 1, ..., OB^{(1)},\ j = 1, ..., OI^{(1)}. \tag{3.44}$$

The following constraints ensure that a class cannot use the same instructor until the previous class is finished. The first constraint serves as an initialization.

$$OC_{l11}^{(1)} - OC_{l01}^{(1)} - OP_l^{(1)} + G(1 - OX_{l11}^{(1)}) \geq 0, \quad \forall\, l = 1, ..., O\sigma. \tag{3.45}$$

$$OC_{lej}^{(1)} - OC_{le'j}^{(1)} - OP_l^{(1)} + G(1 - OX_{lej}^{(1)}) \geq 0, \quad \forall\, e = 2, ..., OB_l^{(1)},\ e' = 1, ..., OB_l^{(1)},\ e' < e \quad J = 1, ..., OI_l^{(1)},\ l = 1, ..., O\sigma. \tag{3.46}$$

In order to ensure that the classes are numbered with respect to their completion times such that the last class has the latest completion time, the following constraint is used.

$$OC_{lej}^{(1)} - OC_{le'j'}^{(1)} + G(1 - OX_{lej}^{(1)}) \geq 0, \quad \forall\, e = 2, ..., OB_l^{(1)},\ e' = 1, ..., OB_l^{(1)},\ e' < e \quad j = 1, ..., OI_l^{(1)}, j' = 1, ...OI_l^{(1)},\ l = 1, ..., \sigma. \tag{3.47}$$

With the current set of constraints, the model has many alternative optimal schedules for choosing instructors and forming classes. These alternative optimal schedules can be reduced by using symmetry breaking constraints by numbering the instructor and picking

them in the increasing order as follows:

$$\sum_{e=1}^{OB_l^{(1)}} OX_{le(j-1)}^{(1)} \geq \sum_{e=1}^{OB_l^{(1)}} OX_{lej}^{(1)}, \quad \forall\ j = 2, ..., OI_l^{(1)},\ l = 1, ..., O\sigma. \tag{3.48}$$

The following constraints allow the classes of variable sizes to be performed while ensuring that their completion times are at least greater than the processing time of BOLC B plus the times at which the soldiers in respective classes enter the system.

a. First Sublot

$$OC_{l1j}^{(1)} - OP_l^{(1)} - OR_o + G(1 - OZ_{ol1j}^{(1)}) \geq 0, \quad \forall\ o \in o_l,\ l = 1, ..., O\sigma,\ j = 1, ..., OI^{(1)}; \tag{3.49}$$

b. For sublots $2, ..., OB_l^{(1)}$

$$OC_{lej}^{(1)} - OP_l^{(1)} - OR_o + G(1 - OZ_{olej}^{(1)}) \geq 0, \quad \forall\ o \in o_l,\ e = 1, ..., OB_l^{(1)},\ j = 1, ..., OI_l^{(1)},\ l = 1, ..., O\sigma. \tag{3.50}$$

Unit Assignment

After the officers complete training in the BOLC B stage, they are assigned to an available BCT. There is no processing that takes place during this stage; however, the BCT requirements must be fulfilled during each window. A binary variable is used to keep track of the BCT to which an officer is assigned. An officer can be assigned to only one BCT:

$$\sum_{b=1}^{B} OZ_{olb}^{(3)} = 1, \quad \forall\ o \in o_l,\ l = 1, .., O\sigma. \tag{3.51}$$

The number of officers assigned to an ARFORGEN BCT must meet the manning

requirement:

$$\sum_{o \in o_l} OZ^{(3)}_{olb} = OReq_{lb}, \quad \forall\ l = 1, ..., O\sigma,\ b = 1...B. \tag{3.52}$$

Besides the ARFORGEN BCT units, the Army must also fulfil demand requirements for both the Generating Force and Operating Force. Unlike the ARFORGEN demand, the time requirements of fulfilling the rest of the Army demand are not as severe. Therefore, the following two constraints allow for shortages of officers for some units only to be fulfilled later with an associated penalty cost in the objective function. The constraints also allow for additional officers to carry over to future unit demands. The first of the two constraints listed sets up the initial excess of officers as zero and the second constraint sets up how the $O\alpha^+_{lb}$ and $O\alpha^-_{lb}$ values track the excesses and shortages of officers graduating BOLC B.

$$O\alpha^+_{l(\beta+1)} - O\alpha^-_{l(\beta+1)} = \sum_{o \in o_l} Z^{(3)}_{ol(\beta+1)} - OReq_{l(\beta+1)}, \quad \forall\ l = 1, ..., O\sigma. \tag{3.53}$$

$$O\alpha^+_{lb} - O\alpha^-_{lb} = O\alpha^+_{l(b-1)} - O\alpha^-_{l(b-1)} + \sum_{o \in o_l} Z^{(3)}_{olb} - OReq_{lb}, \quad \forall\ l = 1, ..., O\sigma,\ b = \beta + 2...B. \tag{3.54}$$

The symmetry breaking constraints for the third stage, or the unit assignment stage, ensure that an officer is assigned to the first available unit in order to reduce the other symmetric optimal solutions.

$$\sum_{b=1}^{B} BW_b(OZ^{(3)})_{olb} \leq \sum_{b=1}^{B} BW_b(OZ^{(3)}_{(o+1)lb}), \quad \forall\ o \in |o_l| - 1,\ l = 1, ..., \sigma. \tag{3.55}$$

Where $|o_l|$ is the number of officers in o_l. The constraints for the binary variable $OY^{(3)}_{olejb}$, which is a product of $OZ^{(1)}_{olej}$ and $OZ^{(3)}_{olb}$, can be linearized using the following inequalities:

$$\begin{aligned} OZ^{(1)}_{olej} &\geq \sum_{b=1}^{B} OY^{(3)}_{olejb}, \\ \forall\ o \in o_l,\ e = 1, ..., OB^{(1)}_l,\ j = 1, ..., OI^{(1)}_l,& \\ l = 1, ..., O\sigma,\ b = 1, ..., B.& \end{aligned} \tag{3.56}$$

$$\begin{aligned} OZ^{(3)}_{olb} &\geq \sum_{e=1}^{OB^{(1)}_l} \sum_{j=1}^{OI^{(1)}_l} OY^{(3)}_{olejb}, \\ \forall\ o \in o_l,\ e = 1, ..., OB^{(1)}_l,\ j = 1, ..., OI^{(1)}_l,& \\ b = 1, ..., B,\ l = 1, ..., O\sigma.& \end{aligned} \tag{3.57}$$

$$\begin{aligned} OZ^{(1)}_{olej} + OZ^{(3)}_{olb} - 1 &\leq OY^{(3)}_{olejb}, \\ \forall\ o \in o_l,\ e = 1, ..., OB^{(1)}_l,\ j = 1, ..., OI^{(1)}_l,& \\ l = 1, ..., O\sigma,\ b = 1, ..., B.& \end{aligned} \tag{3.58}$$

The time an officer waits to be assigned to a unit is determined as follows:

$$\begin{aligned} BW_b - OC^{(1)}_{lej} - G(1 - OY^{(3)}_{olejb}) &\leq OW^{(3)}_o, \\ \forall\ o \in o_l,\ e = 1, ..., OB^{(1)}_l,\ j = 1, ..., OI^{(1)}_l,\ l = 1, ..., O\sigma,\ b = 1, ..., B.& \end{aligned} \tag{3.59}$$

$$\begin{aligned} OW^{(3)}_o &\geq 0 \\ \forall\ o \in o_l,\ l = 1, ..., O\sigma.& \end{aligned} \tag{3.60}$$

The officers meeting the manning requirement of the BCT must also complete BOLC within the window the BCT is given to man units:

$$EW_b - OC^{(1)}_{lej} + G(1 - OY^{(3)}_{olejb}) \geq 0, \quad \forall\ o \in o_l,\ e = 1, ..., OB^{(1)},\ j = 1, ..., OI^{(1)},\ b = 1, ..., B,\ l = 1, ..., O\sigma. \tag{3.61}$$

3.2 Complexity of the AFGP

In this section, we present the complexity of the AFGP. The AFGP is comprised of three problems of varying complexities belonging to stages BT, AIT, and BCT. If any of these problems is NP-Hard, then the AFGP is NP-Hard as well. We show this next.

Proposition 3.2.1. *The AFGP is NP-Hard.*

Proof. In order to determine the overall complexity of the AFGP, we analyze the complexity of each of the problems involved at the BT, AIT and BCT stages.

At the BT stage, the problem of allocating soldiers with dynamic arrivals to the BT classes is a simplified assignment problem because the size of each BT class in unbounded.

At the AIT stage, the problem of assigning the soldiers of a skill set to their classes is a generalized assignment problem with the following variations: (i) the soldiers are available for assignment at different times (depending upon their completion times at Stage 1); (ii) As a result of (i), a soldier can be assigned to any AIT class that starts only after his/her completion time at Stage 1, (iii) the class size for an AIT class is constrained to be between the given upper and lower limits. Variations (i) and (ii) maintain the underlying structure of the generalized assignment problem albeit they reduce combinations of soldiers and the classes that they can be assigned to. However, variation (iii) adds another constraint to the generalized assignment problem. Therefore, the second stage scheduling problem is at least as hard as the generalized assignment problem. Since the generalized assignment problem is NP-Hard [63], so is this problem.

The problem of assigning soldiers to the units at Stage 3 is also a generalized assignment problem with the following variations: (i) the soldiers are available for assignment to units at various times, and (ii) each unit requires fulfillment of an exact number of soldiers of various skill sets. Variation (ii) adds another restriction on the generalized assignment problem because of the requirement of soldiers of various skill sets in one unit. In addition, the formation of Army units permits excesses and shortages of soldiers to its requirements, which act as additional constraints for the generalized assignment problem. Thus, this problem is also at least as hard as the generalized assignment problem. Hence, the AFGP is NP-Hard. □

3.3 Performance of AFGP

We solved the above model directly by using CPLEX 12.1 on an E3535 2 Ghz 3.25G RAM quad core computer, Table 3.3 displays results for small-sized scenarios. The two parameters that we varied for the sample tests are the number of skill sets and the number of soldiers for each instance of skill sets. For two skill sets, the largest model that can be run before the computer runs out of memory contains a total of ten soldiers. As the number of skill sets increases, the model actually runs faster than that with lesser number of skill sets, for the same number of soldiers. For example, for ten soldiers, the two-skill-set case required a run time of almost 25 minutes while for three skill sets it required only about 4.5 minutes. Even more impressive is the reduction in time for the four and five skill set scenarios. The same ten soldier case for the four skill sets took less than 30 seconds to complete and less than 5 seconds for the five skill set case. The reduction in CPU time as the skill sets increase reveals an inherent structural property of the problem, that is, it can be efficiently decomposed through the separability of the skill sets. This is intuitive since across different skill sets, completely different instructors and resources are used at Stage 2, thereby enabling the overall population of soldiers to be decomposed into smaller groups based on their skill

Table 3.1: Solution of model AFGP using CPLEX 12.1 when number of classes per skill set is 4 and number of instructors per skill set is 3

Number of Skill Sets	Number of Soldiers	CPU Time (secs)	Flow Time
2	5	4.39	30.5
2	10	1,496.34	61.9
2	15	out of memory	
2	20	out of memory	
3	5	2.90	42
3	10	91.48	63.7
3	15	out of memory	
3	20	out of memory	
4	5	2.71	51
4	10	28.68	71.2
4	15	out of memory	
4	20	out of memory	
5	5	2.71	58.5
5	10	4.29	79.7
5	15	193.27	104.4
5	20	out of memory	

sets. Therefore, when class times are determined at this stage, far fewer soldiers per skill set have to be considered compared to the overall population. As the results from Table 3.3 indicate, the model presented in this chapter cannot handle reasonably-sized scenarios because of excessive memory requirements. Therefore, we explore other solution approaches in the following chapter.

Chapter 4

Solution Methodology and Analysis of the Model

4.1 Decomposition and Column Generation

As discussed in Chapter 3, the complexity of Model AFGP leaves it intractable for reasonably-sized problems, let alone any scenario that is scaled closer to realistic conditions. In order to improve solvability of the model, the use of optimization techniques for large-scale or specially-structured models is appropriate [5]. Some common techniques are the Dantzig-Wolfe decomposition [5], Benders [5] decomposition, and Lagrangian Relaxation [5], also referred by Bazaraa *et al.* [5] to as applying the decomposition principle. The basic idea of a decomposition principle as detailed in Bazaraa *et al.*[5] is to break up the problem such that its pieces can be solved individually while everything is linked together through the general constraints or master problem. By partitioning the problem into a set of general and specially structured constraints, the problem can be solved by running a linear program over the two different sets such that the information is passed between the sets until optimality is achieved. As the master problem passes updated cost coefficients to the specially structured constraints or subproblems, the subproblem generates new columns with respect to the updated cost coefficients.

An introduction to the column generation method for linear and integer programming is given by Desrosiers *et al.* [29]. In order to apply column generation, a problem has to be reformulated such that the decision variables are expressed as a convex combination of their extreme points. An effective way to reformulate a problem is to exploit its structural properties that enable solving the problem more efficient. For example, the Dantzig-Wolfe method[29] can exploit a block diagonal structure such that each block within the problem structure decomposes into its own subproblem. Within each subproblem exists a convexity constraint and associated dual variables. As there will now be as many subproblems as blocks in the problem structure, each subproblem is solved in turn. Each subproblem is solved, and then, compared at each iteration to determine its eligibility to improve the objective value. Only when none of the subproblems result in improvements to the overall objective value is the problem solved.

Reformulation of problems in terms of the pricing algorithm or subproblem is the main focus in Vanderbeck and Savelsbergh [71]. They specifically address Dantzig-Wolfe decomposition in mixed integer programming. The premise of their work is based on generating sets, which constitute the working space to search for the solutions to the master problem. By creating a reformulation respective of these generating sets, they are able to consider the integrality restrictions of the original problem.

4.2 Literature Review of Column Generation Techniques for Solving Scheduling Problems

The literature review of lot streaming problems in Chapter 2 included several different research efforts across the different machine shop configurations. Upon closer review of the work reported on lot streaming, to the best of our knowledge, the use of column generation has not been used for the solution of these problems. While lot streaming is essentially a

machine scheduling problem, there have been some research efforts reported on the machine scheduling and lot sizing problems that employ decomposition-based methods. We briefly review work reported on the use of decomposition-based or column generation methods for lot sizing and machine scheduling problems.

As part of a collection of column generation-related papers, Huisman *et al.* [40] discuss how column generation can be combined with Lagrangian relaxation as an alternative method for obtaining tighter lower bounds. By combining Dantzig-Wolfe and Lagrangian relaxation techniques in one approach, they are able to avoid the use of the simplex method by directly applying Lagrangian relaxation to what they call an extended formulation, which is also referred to as a reformulation or master problem.

In a brief overview, Huisman *et al.* [40] describe the main idea of Dantzig-Wolfe decomposition succinctly as reformulating the original problem by replacing the variables in the original problem with a convex combination of extreme points and a linear combination of the extreme rays of the polyhedron respective of the formulation structure in place. Included in this master problem are the complicating constraints that keep the problem from being independent subproblems. With this setup, column generation is used as a way to generate columns only when needed, avoiding the column generation for every extreme point. The choice of which column to generate is determined by solving a pricing problem that calculates the negative reduced costs.

Unlike the Dantzig-Wolfe method, Huisman *et al.* [40] explain that in the Lagrangian relaxation problem, the complicating constraints are considered by dualizing them into the objective function. Instead of using column generation, the Lagrangian relaxation uses a vector of positive multipliers such that the multipliers are optimized through the subgradient method. Despite their reformulation differences, Huisman *et al.* [40] observe that the complicating constraints included in the Dantzig-Wolfe reformulation are the same constraints dualized in the Lagrangian relaxation, the optimal values of the two methods are

the same. They observe that the optimal dual variables for the Dantzig-Wolfe master problem correspond to the optimal multipliers in the Lagrangian relaxation method. With these similarities, solving the two problems is almost equivalent since solving the subproblem is the same problem as solving the Lagrangian relaxation except for a constant. They exploit this relationship between the two methods to develop an improved algorithm.

The art of using decomposition-based methods is recognizing the best way to reformulate the problem based on the nature of the problem, specifically its structure. To help explain their algorithm, Huisman *et al.* [40] use the example of a capacitated lot-sizing problem. They provide the mathematical formulation to the problem and observe that without the complicating capacity constraints across the multiple items, the problem decomposes by each item as an uncapacitated lot-sizing problem. The complicating constraints are kept in the master problem of the reformulation and the problem is expressed in terms of production of each item over the time horizon. They create a set of all extreme point production plans for each item. The convexity constraints are therefore expressed in terms of the production plan of each item of extreme production plans. The costs are also reformulated to be in terms of the sum cost of each plan instead of broken out by cost type. Instead of solving the rest of the problem through pricing subproblems and column generation, they use a combination of Dantzig-Wolfe and Lagrangian relaxation. Using the master problem resulting from the Dantzig-Wolfe reformulation, they bring up the capacity constraint in the objective and create subproblems incorporating the Lagrangian relaxation of the capacity constraint. They determine the optimal dual variable for the convexity constraint for a particular item by solving an easily solvable minimization problem of the constraint's reduced cost. The Lagrangian multipliers are optimized by using the subgradient optimization procedure. Combining the Lagrangian multipliers, they are able to generate new columns for the pricing problem.

While Husiman *et al.* [40] focused on reformulating and solving the capacitated lot-

sizing problem, Van den Akker*et al.* [70] have applied column generation to a machine scheduling problem. Specifically, they focus on a parallel machine scheduling problem where the objective is to minimize the weighted completion time. They reformulate the problem in terms of a machine schedule, where a machine schedule represents the string of jobs assigned to a single machine. In order to indicate when a job is in a particular machine schedule, they introduce a binary constant. Using this indicator constant, the completion time per job per schedule is only defined if the associated indicator constant for the same job and schedule equal one. This particular scheduling problem assumes no idle time between jobs, and therefore, the completion times are just the sum of the processing times of all the previous jobs for a given job j. Therefore, the overall cost of a machine schedule can be defined in terms of constants. They use a binary variable that selects machine schedules such that each machine has a schedule, each machine includes a job exactly once, and the total cost is minimized. Using column generation, they start with an initial set of feasible machine schedules and then, add columns with negative reduced cost. Hence, the pricing algorithm involves finding a machine schedule with minimum reduced cost for machine schedules.

In order to solve the pricing problem, Van den Akker *et al.* [70] present a dynamic programming-based pricing algorithm. The jobs are ordered in an increasing weight to processing time ratio, where the weight is the associated weight of the weighted completion time. Using this ordering of jobs, their pricing algorithm uses a forward recursion such that the jobs added are in the increasing index order. The pricing algorithm determines the minimum reduced cost over all feasible machine schedules for a set of jobs such that the last job has a specified completion time. For a machine schedule with minimum reduced cost, the pricing algorithm considers two states. One state belongs to the case when the machine schedule does not contain the current job, while the other state is when the machine schedule contains the current job. If the job is not in the machine schedule, the best machine schedule is selected respective of all the jobs prior to the current job. In the state where the job is contained in the machine schedule, a feasibility condition is given, which ensures that the

completion time of the job is after its ready time plus processing time but it is prior to its due date. If the feasibility condition is met, then the job gets added to the best machine schedule for the jobs up to the current job that finish at the current time less the job's processing time. The interesting aspect about this pricing algorithm is that the calculation for minimum reduced cost is a function of time, regardless of the problem not using a time-indexed formulation.

Another type of machine scheduling application is presented by Van den Akker *et al.* [69] that uses a time-indexed formulation for a single machine problem. A key benefit of a time-indexed formulation for machine scheduling is the generating of tighter linear programming relaxation bounds that help in providing a good guide to approximate algorithms, while its main disadvantage is its large size. They show how the use of Dantzig-Wolfe decomposition can circumvent dealing with large time-indexed formulations. They present a reformulation of the time-indexed formulation for a single-machine scheduling problem. By applying Dantzig-Wolfe decomposition, the number of constraints of the problem are reduced at the cost increment in the overall number of variables. Since column generation allows the variables to be handled through small subsets, the increased number of variables is a reasonable offset to the reduction of constraints. They infer through their analysis of the polytope representing their solution space, expressed as a convex hall of its extreme points, that understanding exactly what the extreme points represent in the problem helps in determining how the problem should be reformulated. In the time-indexed formulation for the machine scheduling problem presented, the extreme points of the polytope represent the schedules satisfying a particular constraint in the original problem. When the master problem for Dantzig-Wolfe is thus formulated, the convex combination of these extreme points and respective weights that sum to one are used to determine the optimal solution at the completion of the solution method. For their problem, the extreme points represent pseudo-schedules, which meets the capacity constraints but the jobs need not start exactly

once.

With the resulting Dantzig-Wolfe reformulation, the number of variables increase dramatically. Therefore, to find the variables with the minimum reduced cost, Van den Akker *et al.* [69] solve an optimization problem called the pricing problem over all the variables. Intuitively, the pricing problem objective is expressed in terms of cost. Therefore, whatever gets subtracted from the cost should be as large as possible, resulting in a minimum reduced cost. The reduced cost is determined using the dual variables associated with the constraints. The pseudo-schedules can be represented as paths in a network. Through the network representation of the problem, the constraint of job-starts is considered. By setting the arc length to the respective job start such that it is the value of the difference of the cost and its dual variable, the pricing problem is the equivalent of solving the shortest path problem. With this network representation, they are able to solve the pricing problem in $O(nT)$ time using dynamic programming.

Bülbül *et al.* [11] consider a m-machine flow shop scheduling problem while minimizing earliness, tardiness, and intermediate inventory holding costs. They approach the problem by using both Dantzig-Wolfe reformulations and Lagrangian relaxation to develop heuristics to minimize total cost. They present two Dantzig-Wolfe reformulations using column generation to solve the problem, and use Lagrangian relaxation to develop lower bounds. Unlike the subproblems addressed in [69] and [70], the subproblems resulting in the problem addressed in Bülbül *et al.* [11] are NP-hard pricing problems. The issues that they address in their subproblems for the m-machine flow shop are similar to those for our model described in Chapter 3. In the scheduling problem they addressed, they relax machine precedence constraints instead of following the norm of relaxing machine capacity constraints. They explain that, through decomposition, the relaxed problem should be easier to solve but the tradeoff of this benefit is looser bounds. Also, the reformulation results in more difficult subproblems are able to obtain effective solutions because of the tighter formulation. They approach

the problem by developing heuristics that solve the subproblems using an enhanced column generation scheme via Lagrangian relaxation and artificial variables in the linear programming master problem. Even though solutions to the subproblems are only approximated, they provide tight lower bounds which the authors use to create lower bounds for the overall problem.

While our problem maybe configured as a flexible flow shop, the precedence constraints Bülbül *et al.* [11] relax to solve their m-machine flow shop problem is something that we have to address. Also, similar to our problem, Bülbül *et al.* [11] address the issue wait time, unlike the parallel machine setup presented in [70]. They determine the wait time by solving a linear program formulated as a timetabling formulation, where the job processing sequences are fixed. This formulation allows for unequal ready times, which is also a feature of our problem. Noting that this timetabling formulation is not tight enough for their near-optimal feasible integer solution, they present their next formulation that is time-indexed. The key benefit of a time-indexed formulation for their problem is achievement of a strong LP relaxation; however, its biggest disadvantage is the rapidity at which the models grow with the number of jobs and processing times.

In their first Dantzig-Wolfe reformulation of the time-indexed m-machine flow shop problem, Bülbül *et al.* [11] develop an integer programming master problem with an exponential number of variables that represent capacity-feasible schedules for individual machines. While the number of variables representing these capacity-feasible schedules is exponential, the number of possible schedules remains finite. Because of integrality restrictions on completion times, ready times, due dates and processing times. These feasible schedules satisfy ready time constraints, which are the same constraints we have in our problem, that is that a job cannot start on a machine until it is finished with a previous machine. The variable they use for the operational ready time is indexed in terms of the current machine and current job. Similar to the work done in [70], Bülbül *et al.* [11] create a binary variable that chooses a schedule for a machine satisfying the capacity constraints of the machine where

each schedule is defined by a set of completion times. In the reformulation they present, this binary variable is used as the convexity constraint. This binary variable, that represents the selection of a machine schedule is multiplied by each of the costs in the objective to include the holding costs for the first machine, the holding costs associated with a job's wait time, and the earliness and tardiness costs. The linking constraints included in this integer programming master problem are the operation precedence constraints that link the completion times of a job from a previous machine to its completion time on the next machine. The convexity constraint included in the master problem restricts the number of schedules that can be selected for a machine to only one feasible schedule for each machine. The complexity of this reformulation is such that using column generation to solve the LP relaxation of this integer formulation is strongly NP-hard. This complexity leads them to develop a near-optimal solution method instead of developing an optimal algorithm. They explain their approach to solve the linear programming relaxation of this integer formulation approximately, followed by constructing the near-optimal feasible schedules using the approximated LP solution. In order to solve the LP approximation, they use column generation to avoid explicitly accounting for the exponentially many feasible schedules on each machine. The dual to their linear programming master problem is used to find the negative reduced costs of the schedules. The subproblems are different from each other and each is described as strongly NP-hard.

Bülbül *et al.* [11] present an alternative decomposition approach of the problem by formulating the original problem using Lagrangian relaxation. Lagrangian relaxation subproblems are equivalent to the pricing problems previously described for the linear master problem, and therefore, have the same complexity issues. Since the subproblems in the LP reformulation are not solved completely, the subgradient optimization approach cannot be used for the Lagrangian formulation since finding a subgradient in each iteration is not guaranteed. Therefore, instead of solving the Lagrangian relaxation, they use the formulation to find lower bounds on the optimal integer solution for a set of dual variables computed from

the subproblems and thus improve the overall performance of their algorithm.

In the second Dantzig-Wolfe reformulation presented, they drop the waiting time variables and instead consider the effect of waiting times for the schedules by adjusting the objective function coefficients. By considering wait-time costs in the objective function of the original problem, the wait time is considered in the pricing problems. Since the wait time is taken into consideration in the objective function, the waiting time is omitted in the precedence constraint, setting the constraint as an inequality to account for a nonzero difference in completion times for a job on a previous and current machine. The pricing problems of this second reformulation differ from the first formulation only by the objective functions. The main difference between the two Lagrangian formulations of the relaxed Dantzig-Wolfe reformulations is that the second has unrestricted dual variables since the operation precedence constraint is now an inequality, resulting from the dropping of the waiting time variable. The resulting pricing problems for both linear relaxed programs of the two Dantzig-Wolfe reformulations are solved through an observation that the coefficients of the completion time and tardiness variables in the objective function are not always nonnegative. A two-phase approach using artificial variables is used to solve the two linear relaxed Dantzig-Wolfe formulations in an effort to bound the pricing problem objectives. This approach is shown to speed up the convergence of the column generation algorithm. The column generation procedure is able to terminate faster because of the calculation of the lower bounds on the linear program master problem . By duality theory, the optimal objective value of the LP master problem can be bounded by the objective value of any dual feasible solution. Thus, using the optimal objective values of the pricing problems, they are able to construct a dual feasible solution, incidentally creating a lower bound on the LP master problem optimal solution.

4.3 Reformulation of Model AFGP

The natural decomposition of a flow shop problem similar to the problem discussed in [11] is by machine stage. While our problem is more similar to a flexible flow shop than the problem discussed in [11], ideally the problem should decompose into three subproblems respective of the three stages: Basic Training, Advanced Individual Training, and formation of Brigade Combat Teams (or Unit formations). The master problem is composed of the convexity constraints that select the optimal set of schedules across all iterations for each stage and the linking constraints that tie the three stages together. Specifically, the linking constraints coordinate the flow of soldiers from one stage to the next. Each subproblem generates a schedule that reduces cost for its respective stage at each iteration. Each schedule generated gives feasible completion times for each soldier in the respective stage. When classes form at the first stage, one class could include several soldiers each with different ready times. Because of this feature, the use of processing and ready times as bounds for the completion times at the second stage are ineffective in producing all feasible schedules.

To ensure generation of all feasible schedules in the subproblems, the BT and AIT stages must be solved as one combined subproblem (designated as ZPP12). The second subproblem (designated as ZPP3) will constitute formation of BCT at the third stage. While ZPP12 is the combination of BT and AIT, the two stages generate schedules differently as they are different in nature. Since BT is the equivalent of a single uncapacitated machine problem, each schedule generated for BT consists of the total number of classes offered that reduces cost, where the cost is measured as waiting time. Each possible number of classes results in a different class configuration and soldier distribution among the varying number of classes. An allocation variable for each stage, $\boldsymbol{y}$, keeps track of the soldier to class allocations for various schedule configurations, j. Therefore, for each schedule configuration, j, an optimal BT schedule with classes $i = 1, ..., j$ is determined for each $j = 1, ..., B^{(1)}$ where $B^{(1)}$ is the maximum number of classes formed in BT. For each iteration, the class

configuration, j, with the most reduced cost is selected as the schedule that enters the basis.

The schedules for AIT are generated differently than the method used for BT. Since AIT is a parallel machine problem, the number of instructors and classes must be determined. Thus, instead of determining which class configuration is best, AIT schedules generate the instructor configuration that gives the most reduced cost for $a_l = 1, ..., I_l^{(2)}$, where $I_l^{(2)}$ is the maximum number of instructors allowed for skill set $l = 1, ..., \sigma$. This schedule generation approach is the same for ever skill set. As one combined subproblem, ZPP12 generates each soldier's schedule for both BT and AIT that results in the greatest reduction of waiting time (cost) at each iteration.

The third stage subproblem, ZPP3, generates the best assignment given the dual values passed from the master problem. This subproblem generates an assignment schedule that ensures that every soldier is assigned to a unit. The linking constraint in the master problem makes sure that whichever BCT assignment schedule is selected, it considers the completion time of the soldier in the previous stage. The following sections present the notation for the reformulation, the reformulation itself, and the pricing problems.

4.3.1 Notation

Parameters

ES	Total number of enlisted soldiers
R_s	Ready time of soldier s before BT
σ	Total number of MOS
T	Total number of iterations of all possible schedules for BT, AIT and BCT
$C_{st}^{(1)}$	Completion time in BT of soldier s in iteration t
$C_{st}^{(2)}$	Completion time in AIT of soldier s in iteration t
$C_{st}^{(3e)}$	Beginning window time to which soldier s is assigned in iteration t during unit assignment
$C_{st}^{(3f)}$	Ending window time to which soldier s is assigned in iteration t during unit assignment

The intent of $C_{st}^{(3e)}$ and $C_{st}^{(3f)}$ will become clear in the sequel. These are the products

of the beginning and ending window times and the assignment variable, which determine assignments made in the subproblem.

Variables

$w_s^{(2)}$ The time an enlisted soldier waits to start AIT, $\forall\ s = 1, ..., ES$.
E_s Earliness of soldier s assignment to BCT.
T_s Tardiness of soldier s assignment to a unit (Tardiness is not permitted for a BCT).

$$x_t = \begin{cases} 1, & \text{if schedule } t \text{ is selected} \\ 0, & \text{otherwise, } \forall\ l = 1, ...j. \end{cases}$$

$$x_t^{(3)} = \begin{cases} 1, & \text{if schedule } t \text{ is selected} \\ 0, & \text{otherwise, } \forall\ l = 1, ...j. \end{cases}$$

The above notation for the reformulated model is not as extensive as that for the model presented in Chapter 3. The reformulated model using the binary $x_t,\ t = 1, ..., T$ indicator variables allows the completion times to be determined in the pricing problems, which are therefore parameters in the master problem (presented in the following section). The allocation $\boldsymbol{y}$ values for each stage are also determined in the pricing problems respective of the appropriate stage, and these allocation values are also parameters in the master problem. Most of the calculations for this reformulated model occurs within each of the pricing problems that generate reduced costs, feasible schedules, and allocation of soldiers for each stage. The linking constraints are considered through the use of dual variables passed to the pricing problems. Therefore, we relax the binary x_t values to be linear in order to generate the necessary dual variables needed to pass to the appropriate pricing problems.

Consider the following reformulated model, designated as model AFGP-R. It is developed only for enlisted soldiers. As noted earlier, the enlisted soldiers and officers do not share resources, and hence, can be treated separately.

4.3.2 Model AFGP-R

Minimize:

$$\sum_{l=1}^{\sigma}\sum_{s\in s_l}\left(\sum_{t\in T}(-P^{(1)}-R_s+C_{st}^{(2)}-P_l^{(2)})x_t\right)+\sum_{l=1}^{\sigma}\sum_{s\in s_l}(E_s+T_s) \tag{4.1}$$

Subject to:

Enlisted Basic Training and Individual Training

$$(\gamma^{(1)}) \quad \sum_{t\in T} x_t = 1. \tag{4.2}$$

Enlisted Unit Formation

$$(\gamma^{(3)})\sum_{t\in T} x_t^{(3)} = 1. \tag{4.3}$$

Complicating Constraints

$$(\mu_s^{(2)}) \quad \sum_{t=1}^{T} C_{st}^{(2)}x_t - \sum_{t=1}^{T} C_{st}^{(3e)}x_t^{(3)} = w_s^{(e3^-)} - E_s, \quad \forall\ s\in s_l,\ l=1,...,\sigma. \tag{4.4}$$

$$(\mu_s^{(3)}) \quad \sum_{t=1}^{T} C_{st}^{(2)}x_t - \sum_{t=1}^{T} C_{st}^{(3f)}x_t^{(3)} = T_s - w_s^{(t3^+)}, \quad \forall\ s\in s_l,\ l=1,...,\sigma. \tag{4.5}$$

$$(\mu_s^{(4)}) \quad \sum_{t=1}^{T} C_{st}^{(2)}x_t - \sum_{t=1}^{T} C_{st}^{(3f)}x_t^{(3)} = 0, \quad \forall\ s\in s_{\beta_l},\ l=1,...,\sigma. \tag{4.6}$$

The objective function shown in Equation (4.1), is separated into the combined first

and second stage waiting times and the third stage earliness and tardiness. Overall, the objective is to minimize flow time. Since our processing times are fixed parameters, we can minimize flow time by minimizing wait time. The first expression in (4.1) is essentially the sum over soldier wait times between the soldier arrival and BT class start time and between the BT class finish and AIT class start times. The completion times for soldiers are different given a schedule t, and therefore, depend on the selection of a schedule. Hence, the quantity is multiplied by x_t, the schedule indicator for the first and second stage set of schedules. The earliness and tardiness variables are determined in constraints (4.4) and (4.5) such that the completion and allocation values associated with the schedules selected are considered in the calculations.

Within each stage, there are constraints limiting the number of schedules that can be selected for each stage. Constraint (4.2) restricts the BT and AIT stage to only one schedule. Constraint (4.3) enforces the selection of one of the possible BCT schedules generated.

The last set of constraints, (4.4) and (4.5), are the linking constraints that handle the constraints that include parameters and variables across different stages. The constraint set (4.4) ensures that a soldier cannot be assigned to a unit before the completion of the AIT class subject to being assigned before the beginning window of the unit or assigned after. Likewise, the constraint set (4.5) prohibits a soldier from being assigned to a unit after the ending window subject to a tardiness variable. Note that $C_{st}^{(3e)}$ is the product of the beginning window of the unit a soldier s is assigned to and the solider assignment indicator value $y_{slb}^{(3)}$ for a particular t, and similarly $C_{st}^{(3f)}$ is the product of the ending window of the unit the solider is assigned to and the indicator value $y_{slb}^{(3)}$ for a particular t. Using these product values are necessary because the units are not indexed in the master problem specifically, and indicating units causes complexities in column composition.

The constraints of the dual problem give the associated reduced cost of the variables. Below, the dual to our reformulation is presented to explicitly demonstrate determination of

the objectives for each of the pricing problems. The two different constraints are related to the the x_t variables associated with each stage.

Dual

Maximize:

$$\gamma^{(1)} + \gamma^{(3)} \tag{4.7}$$

Subject to:

$$(x) \quad \sum_{l=1}^{\sigma} \sum_{s \in s_l} \left(\mu_s^{(4)} + \mu_s^{(3)} + \mu_s^{(2)}\right) C_{st}^{(2)} + \gamma^{(1)} \leq \sum_{l=1}^{\sigma} \sum_{s \in s_l} (C_{st}^{(2)} - P_l^{(2)}) + \sum_{l=1}^{\sigma} \sum_{s \in s_l} (-R_s - P^{(1)}), \quad \forall\, t = 1, ..., T. \tag{4.8}$$

$$(x^{(3)}) \quad -\sum_{l=1}^{\sigma} \sum_{s \in s_l} C_{st}^{(3e)} \mu_s^{(2)} - \sum_{l=1}^{\sigma} \sum_{s \in s_l} C_{st}^{(3f)} \mu_s^{(3)} + \gamma^{(3)} \leq 0, \quad \forall\, t = 1, ..., T. \tag{4.9}$$

$$\mu_s^{(2)}\ \mu_s^{(3)},\ \gamma^{(1)},\ \gamma^{(3)} \text{unrestricted} \tag{4.10}$$

Note that $C_{st}^{(3e)} = BW_b y_{slb}^{(3)}$ and $C_{st}^{(3f)} = EW_b y_{slb}^{(3)}$.

4.3.3 Pricing Problems

In order for the completion times and allocation values to be parameters in the master problem, they are calculated in the pricing problems respective of the stages. The objective values of each pricing problem is the reduced cost or the dual of the associated x_t variable of the master problem. The constraints of each of the pricing problems include the constraints ensuring feasibility in determining the completion times, allocation values, and soldier unit assignments for each of the pricing problems.

Notation

Parameters

ES	Total number of enlisted soldiers
R_s	Ready time of soldier s before BT
σ	Total number of MOS
$U^{(2)}$	Maximum sublot size in AIT based on instructor-student ratio
$L^{(2)}$	Minimum sublot size in AIT based on instructor-student ratio
$B^{(2)}$	Upperbound of the number of classes for AIT
Req_{lb}	The enlisted personnel of skill l required for BCT or Army unit b, $\forall l = 1, ..., \sigma,\ b = 1, ..., B$
$\gamma^{(1)}$	Dual variable for constraint (4.2)
$\gamma^{(3)}$	Dual variable for constraint (4.3)
$\mu_s^{(2)}$	Dual variable for constraint (4.4)
$\mu_s^{(3)}$	Dual variable for constraint (4.5)

Variables

$C_{si}^{(1)}$	Completion time of class i from Basic Training, $\forall\ i = 1, ..., B^{(1)}$
$C_{slak}^{(2)}$	Completion time for skill set l of instructor a from AIT, $\forall\ s \in s_l,\ a = 1, ..., A_l,\ l = 1, ..., \sigma,\ k = 1, ..., B^{(2)}$
$C_{st}^{(3e)}$	Beginning time of window to which soldier s is assigned during iteration t, $\forall s \in s_l,\ l = 1, ..., \sigma,\ t = 1, ..., T$
$C_{st}^{(3f)}$	Ending time of window to which soldier s is assigned during iteration t, $\forall s \in s_l,\ l = 1, ..., \sigma,\ t = 1, ..., T$

$$y_{si}^{(1)} = \begin{cases} 1, & \text{if soldier } s \text{ is in class } i \\ 0, & \text{otherwise, } \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., j \end{cases}$$

$$y_{slak}^{(2)} = \begin{cases} 1, & \text{if soldier } s \text{ with instructor } a \text{ in class } k \\ 0, & \text{otherwise, } \forall\ s \in s_l,\ a = 1, ..., I^{(2)},\ l = 1, ..., \sigma, \\ & k = 1, ..., B^{(2)}. \end{cases}$$

$$y_{slb}^{(3)} = \begin{cases} 1, & \text{if soldier } s \text{ is in unit } b \\ 0, & \text{otherwise, } \forall\ s \in s_l,\ l = 1, ..., \sigma,\ b = 1, ..., B. \end{cases}$$

ZPP12:

Minimize

$$\sum_{l=1}^{\sigma}\sum_{s\in s_l}\sum_{a=1}^{A_l}\sum_{k=1}^{B^{(2)}} C_{sak}^{(2)} - (P_l^{(2)}) + \sum_{a=1}^{A_l}\sum_{k=1}^{B^{(2)}}(-\mu_s^{(4)} - \mu_s^{(3)} - \mu_s^{(2)})C_{sak}^{(2)} + \sum_{l=1}^{\sigma}\sum_{s\in s_l}\left(-R_s - P^{(1)}\right) - \gamma^{(1)} \tag{4.11}$$

Subject to:

$$\sum_{i=1}^{j} y_{si}^{(1)} = 1, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{4.12}$$

$$C_{si}^{(1)} \leq (P^{(1)} + R_s + ED)y_{si}^{(1)}, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., j. \tag{4.13}$$

$$C_{si}^{(1)} - (P^{(1)} + R_s)y_{si}^{(1)} \geq 0, \quad \forall\ i = 1, ..., j,\ s \in s_l,\ l = 1, ..., \sigma. \tag{4.14}$$

$$C_{s'i}^{(1)} - C_{si-1}^{(1)} - P^{(1)} + G(1 - y_{s'i}^{(1)}) \geq 0, \quad \forall\ i = 2, ..., j,\ s' \in s_l,\ s \in s_l,\ l = 1, ..., \sigma\ s' \neq s. \tag{4.15}$$

$$C_{s'i}^{(1)} - C_{si}^{(1)} + G(1 - y_{s'i}^{(1)}) + G(1 - y_{si}^{(1)}) \geq 0 \quad \forall\ i = 1, ..., j,\ s' \in s_l,\ s \in s_l,\ l = 1, ..., \sigma\ s' \neq s. \tag{4.16}$$

Advanced Individual Training Stage

Class Assignment:

$$\sum_{a=1}^{A_l}\sum_{k=1}^{B^{(2)}} y_{sak}^{(2)} = 1, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{4.17}$$

Class Minimum Size:

$$\sum_{s \in s_l} y^{(2)}_{sak} \geq z_{ak} min_l, \quad \forall\ a = 1, ..., A_l,\ k = 1, ..., B^{(2)},\ l = 1, ..., \sigma. \tag{4.18}$$

Class Maximum Size:

$$\sum_{s \in s_l} y^{(2)}_{sak} \leq z_{ak} max_l, \quad \forall\ a = 1, ..., A_l,\ k = 1, ..., B^{(2)},\ l = 1, ..., \sigma. \tag{4.19}$$

Class Order Precedence:

$$z_{ak} \leq z_{a(k-1)}, \quad \forall\ a = 1, ..., A_l,\ k = 1, ..., B^{(2)},\ l = 1, ..., \sigma. \tag{4.20}$$

Class Schedule Feasibility:

$$C^{(2)}_{sak} - (P^{(2)}_l + C^{(1)}_{si}) + G(1 - y^{(1)}_{si}) + G(1 - y^{(2)}_{sak}) \geq 0, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ a = 1, ..., A_l,\ k = 1, ..., B^{(2)},\ i = 1, ..., j. \tag{4.21}$$

Class Precedence:

$$C^{(2)}_{s'ak} - C^{(2)}_{sak-1} - P^{(2)}_l + G(1 - y^{(2)}_{s'ak}) \geq 0, \quad \forall\ a = 1, ..., A_l,\ k = 1, ..., B^{(2)},\ s' \in s_l,\ s \in s_l\ s' \neq s, \quad l = 1, ..., \sigma. \tag{4.22}$$

Class Times Equality:

$$C^{(2)}_{s'ak} - C^{(2)}_{sak} + G(1 - y^{(2)}_{s'ak}) + G(1 - y^{(2)}_{sak}) \geq 0, \quad \forall\ a = 1, ..., A_l,\ k = 1, ..., B^{(2)},\ s' \in s_l,\ s \in s_l\ s' \neq s, \quad l = 1, ..., \sigma. \tag{4.23}$$

Completion Times:

$$C^{(2)}_{sak} \leq G(y^{(2)}_{sak}), \quad \forall\ s \in s_l,\ a = 1, ..., A_l,\ k = 1, ..., B^{(2)}, \quad l = 1, ..., \sigma. \tag{4.24}$$

This first pricing problem stipulates the feasibility conditions for completion times for BT classes and the assignment soldiers to classes. Constraint set (4.12) enforces a soldier to be assigned to at least one class i for schedule j. The next constraint, (4.13), restricts the waiting time between arrival time of the soldier and start time of the BT class to which he/she is assigned to be less than a fixed parameter, ED. The following constraint set, (4.14) enforces the completion time for any soldier s to be at least greater than the processing time plus the available time. Constraint set (4.15) ensures that a class for any soldier can begin only after a previous class has been completed. Since these completion times are now expressed in terms of soldiers instead of just classes, constraint set (4.16) makes sure that the completion times for the soldiers in the same class are the same. As is the case with other pricing problems, these subproblems are defined for each schedule j that gives the most reduced cost, and the respective column is then passed back to the master problem.

The constraints for the AIT stage in ZPP12 are similar to those the same stage in the model presented in Chapter 3, albeit less complicated. While the subproblem determines the number of AIT instructors necessary for each skill set, the model may solve a one or more instructor model in a single iteration. Since, we only send the completion time per soldier to the master problem, a lot of the complicated class constraints in the model presented in Chapter 3 are unnecessary for this reformulation. Expressing the completion times in terms of soldiers instead of classes simplifies the model. Defining the completion time in terms of soldier works better in the reformulation than in the original model since the stages are decomposed.

ZPP3:

The unit assignment stage includes all the constraints that make sure that the unit requirements are met by the soldiers finishing classes at the AIT stage. The only time-related entities are the beginning and ending window parameters in the objective of the third pricing problem. The timing constraints for this problem are solely accounted for by

the linking constraints in the master problem. We have, $\forall\, t = 1, ..., T$,

Minimize

$$\left(\sum_{b=1}^{B}\sum_{l=1}^{\sigma}\sum_{s\in s_l} BW_b\mu_s^{(2)}y_{sbt}^{(3)}\right) + \left(\sum_{b=1}^{\beta}\sum_{l=1}^{\sigma}\sum_{s\in s_l} EW_b\mu_s^{(3)}y_{sbt}^{(3)}\right) + \left(\sum_{b=\beta+1}^{B}\sum_{l=1}^{\sigma}\sum_{s\in s_l} EW_b\mu_s^{(4)}y_{sbt}^{(3)}\right) - \gamma^{(3)} \tag{4.25}$$

Subject to:

$$\sum_{b=1}^{B} y_{sbt}^{(3)} = 1, \quad \forall\, s \in s_l,\ l = 1, ..., \sigma. \tag{4.26}$$

$$\sum_{s\in s_l} y_{sbt}^{(3)} = Req_{lb}, \quad \forall\, l = 1, ..., \sigma,\ b = 1, ..., \beta. \tag{4.27}$$

$$\sum_{s\in s_l} y_{sbt}^{(3)} \geq Req_{lb} \quad \forall\, l = 1, ..., \sigma,\ b = \beta + 1, ..., B. \tag{4.28}$$

The constraint set (4.26) ensures that every soldier is assigned to some unit. The constraints sets (4.59) and (4.60) require that the manning requirements are met. The difference between the requirement satisfaction for ARFORGEN and Army units is that the requirement for the ARFORGEN units must be satisfied exactly while that for the Army is allowed to have more than what the requirements are. No shortages are allowed for the Army units using these constraints; however, the soldiers can arrive to the units tardy to fulfill the Army requirements.

4.4 Performance of AFGP-R

Compared with the model presented in Chapter 3, this reformulation of the model using Dantzig-Wolfe decomposition demonstrates significant improvements. As shown in 3.3, the model from Chapter 3 could only solve very small problems before running out of memory. In our experimental investigation, some of the same scenarios that timed-out in this more complicated model, the reformulation solves them in a reasonable amount of time.

4.5 Improvement of AFGP-R

Although the model resulting from the reformulation of the problem discussed above does perform better than the original model, we can further exploit its structural properties to enhance its computational efficiency. This pertains to the method of enumerating over the possible schedules for the first two stages in order to determine a column that gives the most reduced cost. Since the first and second stage problems are solved in a single combined subproblem, all the different schedules for the first and second stages must be enumerated. As an example, if there is a maximum of three classes allowed in BT that can be formed and a maximum of two instructors available per two different skill sets, the subproblem must enumerate over all possible combinations to find the column with the most reduced cost. Even though the second stage with its different skill sets breaks down into individual parallel machine-type problems, they are still solved together in the combined subproblem. With all the skill sets mixed in the first stage, the second stage cannot be solved per skill set unless the first stage is solved first. This begs the question: if the first stage problem is solved independently and the completion times are passed to the second stage, will the solution still be optimal, overall? If solving the first stage independently is optimal, then not only will the second stage be able to generate schedules for individual skill sets, but also, will consider a reduced number of schedules necessary to solve the problem. Consider the

following result.

Proposition 4.5.1. *The first stage (BT) problem can be solved to optimality independently without affecting the optimality of the overall problem if all the soldiers arrive at the same time.*

Proof. Let $C_{si}^{(1)*}$ and $C_{sak}^{(2)*}$ be the optimal solution of the subproblem ZPP12. Consider the first stage of ZPP12, and let it be solved independently. Let $\hat{C}_{si}^{(1)}$ be the optimal solution of this partial subproblem. We want to show that $C_{si}^{(1)*} = \hat{C}_{si}^{(1)}$. We have the following cases: (1) $C_{si}^{(1)*} > \hat{C}_{si}^{(1)}$. Then $\hat{C}_{si}^{(1)} + C_{sak}^{(2)*} < C_{si}^{(1)*} + C_{sak}^{(2)*}$, which is a contradiction since $C_{si}^{(1)*}$ and $C_{sak}^{(2)*}$ constitute an optimal solution of ZPP12. (2) $C_{si}^{(1)*} < \hat{C}_{si}^{(1)}$. But, this is not possible because the class size at stage 1 is unbounded, and all soldiers arrive at the same time. Hence, $C_{si}^{(1)*} = \hat{C}_{si}^{(1)}$. □

Note that Proposition 4.5.1 holds true for an un-capacitated machine at Stage 1 and equal ready times. Our formulation of the AFGP problem includes an un-capacitated machine at Stage 1 but uses unequal ready times, therefore, solving the problem sequentially will not be optimal since one of the key assumptions is violated. Given the benefits of reduced total enumeration, we investigate the quality of the solution generated using the sequential solution heuristic.

The following model is similar to the initial reformulation model; however, its biggest distinction is in the number of subproblems. As the first stage becomes its own subproblem, by being solved independently, the second stage becomes a quasi-type subproblem. Since there is a value being passed from the first subproblem to this new second subproblem, it constitutes a deviation from the traditional Dantzig-Wolfe decomposition method. With a separate subproblem, there is also another set of convexity variables shown in the master problem of the reformulation that reflects the AIT schedule selected for each skill set. With this addition, there is also a new linking constraint to ensure that the schedule selected in

the second stage corresponds with the completion time of the first stage.

The notation for this new reformulation is mostly the same as AFGP-R with the additional convexity variables shown here.

$$x_t^{(1)} = \begin{cases} 1, & \text{if schedule } t \text{ is selected} \\ 0, & \text{otherwise.} \end{cases}$$

$$x_t^{(2)} = \begin{cases} 1, & \text{if schedule } t \text{ is selected} \\ 0, & \text{otherwise.} \end{cases}$$

Minimize:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} \left(\sum_{t \in t}^{T} (-P^{(1)} - R_s) x_t^{(1)} + (C_{st}^{(2)} - P_l^{(2)}) x_{tl}^{(2)} \right) + \sum_{l=1}^{\sigma} \sum_{s \in s_l} (E_s + T_s) \tag{4.29}$$

Subject to:

Enlisted Basic Training

$$(\gamma^{(1)}) \quad \sum_{t=1}^{T} x_t^{(1)} = 1. \tag{4.30}$$

Enlisted Advanced Individual Training

$$(\gamma_l^{(2)}) \quad \sum_{t=1}^{T} x_{tl}^{(2)} = 1. \quad \forall\, l = 1, ..., \sigma \tag{4.31}$$

Enlisted Unit Formation

$$(\gamma^{(3)}) \quad \sum_{t \in T} x_t^{(3)} = 1. \tag{4.32}$$

Complicating Constraints

$$(\mu_s^{(1)}) \quad \sum_{t=1}^{T} \left((C_{st}^{(2)} - P_l^{(2)}) x_{tl}^{(2)} - C_{st}^{(1)} x_t^{(1)} \right) = w_s^{(2)}, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{4.33}$$

$$(\mu_s^{(2)}) \quad \sum_{t=1}^{T} C_{st}^{(2)} x_{tl}^{(2)} - \sum_{t=1}^{T} C_{st}^{(3e)} x_t^{(3)} = w_s^{(e3^-)} - E_s, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{4.34}$$

$$(\mu_s^{(3)}) \quad \sum_{t=1}^{T} C_{st}^{(2)} x_{tl}^{(2)} - \sum_{t=1}^{T} C_{st}^{(3t)} x_t^{(3)} = T_s - w_s^{(t3^+)}, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{4.35}$$

$$(\mu_s^{(4)}) \quad \sum_{t=1}^{T} C_{st}^{(2)} x_{tl}^{(2)} - \sum_{t=1}^{T} C_{st}^{(3t)} x_t^{(3)} = 0, \quad \forall\ s \in s_\beta. \tag{4.36}$$

where s_β is the set of soldiers assigned to a BCT.

The objective of this new reformulation shown in expression (4.29) determines the same performance measure as the initial reformulation although defined slightly differently. As there are two subproblems respective to Stage 1 and Stage 2, the original convexity variable x_t splits into two parts, $x_t^{(1)}$ and $x_{tl}^{(2)}$. The convexity variable $x_t^{(1)}$ relates to all schedules generated by Stage 1 for each iteration, t. Likewise the convexity variable $x_{tl}^{(2)}$ relates to the schedules generated from Stage 2 for a skill set l in each iteration t. With the two stages generating independent schedules, the constraint set (4.33) enforces the master problem to only pick Stage 1 and Stage 2 schedules such that every soldier's completion time in Stage 2 is after the soldier's completion time in Stage 1. The remaining linking constraints calculate the earliness and tardiness of a soldier's assignment to a BCT in the third stage. The earliness and tardiness values are determined with respect to the completion time of the

second stage. Therefore, the convexity variable that selects a schedule has the completion time for the second stage as the coefficient.

The dual reflects the modified master problem as shown by the three γ values in the objective function. Each γ represents the $\boldsymbol{x}$ convexity variable by stage. The number of constraints increases by one as each constraint is written as the dual to the corresponding $\boldsymbol{x}$ values in the master problem. With the split of the original subproblem into two subproblems, the terms associated with each constraint are all associated with the same stage as the particular $\boldsymbol{x}$ value (with the exception of the dual variables). To determine the objectives of the pricing problems these three constraints are used as a method to find the most reduced cost.

Dual

Maximize:

$$\gamma^{(1)} + \sum_{l=1}^{\sigma} \gamma_l^{(2)} + \gamma^{(3)}. \tag{4.37}$$

Subject to:

$$(x^{(1)}) \quad -\sum_{l=1}^{\sigma}\sum_{s\in s_l} \mu_s^{(1)} C_{st}^{(1)} + \gamma^{(1)} \leq \sum_{l=1}^{\sigma}\sum_{s\in s_l} (-R_s - P^{(1)}), \quad \forall t = 1, ..., T. \tag{4.38}$$

$$(x^{(2)}) \quad \sum_{l=1}^{\sigma}\sum_{s\in s_l} \left(\mu_s^{(4)} + \mu_s^{(3)} + \mu_s^{(2)}\right) C_{stk}^{(2)} + \sum_{l=1}^{\sigma} \gamma_l^{(2)} \leq \sum_{l=1}^{\sigma}\sum_{s\in s_l} (C_{st}^{(2)} - P_l^{(2)}), \tag{4.39}$$

$$(x^{(3)}) \quad -\sum_{l=1}^{\sigma}\sum_{s\in s_l} C_{st}^{(3e)} \mu_s^{(2)} - \sum_{l=1}^{\sigma}\sum_{s\in s_l} C_{st}^{(3f)} \mu_s^{(3)} + \gamma^{(3)} \leq 0, \quad \forall\, t = 1, ..., T. \tag{4.40}$$

$$\mu_s^{(1)},\ \mu_s^{(2)},\ \mu_s^{(3)},\ \gamma^{(1)},\ \gamma_l^{(2)},\ \gamma^{(3)} \text{unrestricted} \tag{4.41}$$

where $C_{st}^{(3e)} = BW_b y_{slb}^{(3)}$ and $C_{st}^{(3f)} = EW_b y_{slb}^{(3)}$.

Pricing Problems

The first pricing problem generates feasible schedules for only the first stage in this modification. Based on its objective, the pricing problem returns a feasible schedule with most reduced cost (wait time) for the master problem. We pass the completion time of every soldier for Stage 1 to Stage 2. The second stage determines feasible schedules based on the completion time of the first stage and the processing time of the second stage. Now that the second stage can be solved independent of the first stage, each skill set generates its own set of schedules as a subproblem is solved for each skill set separately. The structure of each subproblem is identical, and there are as many subproblems solved at this stage as there are total skill sets.

The new notation for the pricing problems for this improved reformulation of AFGP-R follows.

Notation

Parameters

$\gamma^{(1)}$ Dual variable for constraint 4.30
$\gamma_l^{(2)}$ Dual variable for constraint 4.31
$\gamma^{(3)}$ Dual variable for constraint 4.32
$\mu_s^{(1)}$ Dual variable for constraint 4.33
$\mu_s^{(2)}$ Dual variable for constraint 4.34
$\mu_s^{(3)}$ Dual variable for constraint 4.35

ZPP1:

Minimize

$$\sum_{l=1}^{\sigma}\sum_{s\in s_l}\left(\sum_{i=1}^{t}C_{si}^{(1)}-(R_s+P^{(1)})+\mu^{(1)}\sum_{i=1}^{t}C_{si}^{(1)}\right)-\gamma^{(1)} \tag{4.42}$$

Subject to:

$$\begin{gathered}\sum_{i=1}^{j}y_{si}^{(1)}=1,\\ \forall\ s\in s_l,\ l=1,...,\sigma.\end{gathered} \tag{4.43}$$

$$\begin{gathered}C_{si}^{(1)}\leq(P^{(1)}+R_s+ED)y_{si,}^{(1)}\\ \forall\ s\in s_l,\ l=1,...,\sigma,\ i=1,...,j.\end{gathered} \tag{4.44}$$

$$\begin{gathered}C_{si}^{(1)}-(P^{(1)}+R_s)y_{si}^{(1)}\geq 0\\ \forall\ s\in s_l,\ l=1,...,\sigma,\ i=1,...,j.\end{gathered} \tag{4.45}$$

$$\begin{gathered}C_{s'i}^{(1)}-C_{si-1}^{(1)}-P^{(1)}+G(1-y_{s'i}^{(1)})\geq 0\\ \forall\ i=2,...,j,s'\in s_l,\ s\in s_l,\ l=1,...,\sigma,\ s'\neq s.\end{gathered} \tag{4.46}$$

$$\begin{gathered}C_{s'i}^{(1)}-C_{si}^{(1)}+G(1-y_{s'i}^{(1)})+G(1-y_{si}^{(1)})\geq 0\\ \forall\ i=1,...,j,s'\in s_l,\ s\in s_l,\ l=1,...,\sigma,\ s'\neq s.\end{gathered} \tag{4.47}$$

Note that each iteration t for ZPP1 corresponds to the number of BT classes offered.

ZPP2:

Minimize

$$\sum_{l=1}^{\sigma}\left(\sum_{s\in s_l}\sum_{a=1}^{A_l}\sum_{k=1}^{B^{(2)}}C_{slak}^{(2)}-(P_l^{(2)})+\sum_{s\in s_l}\sum_{a=1}^{A_l}\sum_{k=1}^{B^{(2)}}(-\mu_s^{(4)}-\mu_s^{(3)}-\mu_s^{(2)})C_{slak}^{(2)}-\sum_{s\in s_l}\sum_{a=1}^{A_l}\sum_{k=1}^{B^{(2)}}\mu_s^{(1)}(C_{slak}^{(2)}-P_l^{(2)})\right)$$
$$-\sum_{l=1}^{\sigma}\gamma_l^{(2)} \tag{4.48}$$

Subject to:

Class Assignment:

$$\sum_{a=1}^{A_l} \sum_{k=1}^{B^{(2)}} y_{slak}^{(2)} = 1, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{4.49}$$

Minimum Class Size:

$$\sum_{s \in s_l} y_{slak}^{(2)} \geq z_{ak} min_l, \quad \forall\ a = 1, ..., A_l,\ k = 1, ..., B^{(2)},\ l = 1, ..., \sigma. \tag{4.50}$$

Maximum Class Size:

$$\sum_{s \in s_l} y_{slak}^{(2)} \leq z_{ak} max_l, \quad \forall\ a = 1, ..., A_l,\ k = 1, ..., B^{(2)},\ l = 1, ..., \sigma. \tag{4.51}$$

Precedence Order Army Classes:

$$z_{ak} \leq z_{a(k-1)}, \quad \forall\ a = 1, ..., A_l,\ k = 1, ..., B^{(2)},\ l = 1, ..., \sigma. \tag{4.52}$$

Feasibility of Class Schedule:

$$C_{slak}^{(2)} - (P_l^{(2)} + C_{si}^{(1)}) + G(1 - y_{si}^{(1)}) + G(1 - y_{slak}^{(2)}) \geq 0, \quad \forall\ s \in s_l,\ a = 1, ..., A_l,\ l = 1, ..., \sigma,\ k = 1, ..., B^{(2)},\ i = 1, ..., j. \tag{4.53}$$

Class Precedence:

$$C_{s'lak}^{(2)} - C_{slak-1}^{(2)} - P_l^{(2)} + G(1 - y_{s'lak}^{(2)}) \geq 0, \quad \forall\ a = 1, ..., A_l,\ k = 1, ..., B^{(2)},\ s' \in s_l,\ s \in s_l\ s' \neq s, \quad l = 1, ..., \sigma. \tag{4.54}$$

Equality of Class Times:

$$\begin{aligned} C^{(2)}_{s'lak} - C^{(2)}_{slak} + G(1 - y^{(2)}_{s'lak}) + G(1 - y^{(2)}_{salk}) \geq 0, \\ \forall\ a = 1, ..., A_l,\ k = 1, ..., B^{(2)},\ s' \in s_l,\ s \in s_l\ s' \neq s, \\ l = 1, ..., \sigma. \end{aligned} \tag{4.55}$$

Completion Times:

$$\begin{aligned} C^{(2)}_{slak} \leq G(y^{(2)}_{slak}), \\ \forall\ s \in s_l,\ a = 1, ..., A_l,\ k = 1, ..., B^{(2)}, \\ l = 1, ..., \sigma. \end{aligned} \tag{4.56}$$

ZPP3:

Minimize

$$\left(\sum_{b=1}^{B}\sum_{l=1}^{\sigma}\sum_{s\in s_l} C^{(3e)}_{st}\mu^{(2)}_s\right) + \left(\sum_{b=1}^{B}\sum_{l=1}^{\sigma}\sum_{s\in s_l} C^{(3f)}_{st}\mu^{(3)}_s\right) + \left(\sum_{b=1}^{B}\sum_{l=1}^{\sigma}\sum_{s\in s_l} C^{(3f)}_{st}\mu^{(4)}_s\right) - \gamma^{(3)} \tag{4.57}$$

Subject to:

$$\begin{aligned} \sum_{b=1}^{B} y^{(3)}_{slbt} = 1 \\ \forall\ s \in s_l,\ l = 1, ..., \sigma. \end{aligned} \tag{4.58}$$

$$\begin{aligned} \sum_{s\in s_l} y^{(3)}_{slbt} = Req_{lb} \\ \forall\ l = 1, ..., \sigma,\ b = 1, ..., \beta. \end{aligned} \tag{4.59}$$

$$\begin{aligned} \sum_{s\in s_l} y^{(3)}_{slbt} \geq Req_{bl} \\ \forall\ l = 1, ..., \sigma,\ b = \beta + 1, ..., B. \end{aligned} \tag{4.60}$$

Note that $C^{(3e)}_{st} = BW_b y^{(3)}_{slbt}$ and $C^{(3f)}_{st} = EW_b y^{(3)}_{slbt}$ for iteration t.

Chapter 5

Computational Investigation of the Proposed Methodologies

5.1 Formation of Brigade Combat Teams

In the model formulation of the Army Force Generation Problem, we modeled the Initial Military Training process of Army soldiers followed by their assignment to Brigade Combat Teams. Specifically, we modeled the path of enlisted Army soldiers, those soldiers that enlist into the Army as opposed to being commissioned. Overall, the model includes scheduling Basic Training classes in what we refer to as the first stage. The model then creates schedules for classes in the next stage of IMT entitled Advanced Individual Training. At this point in training, the soldiers separate based on their Military Occupational Speciality. The duration of training and amount of available resources for each MOS during AIT are independent by the MOS. In an effort to model the scenario as realistically as possible, we collected data for the various stages of the IMT and BCT schedule and requirements from Fiscal Year 2009 through Fiscal Year 2010, where an FY is defined as the time between October 1st through September 30th. During this time period, the US Army Recruiting Command reported an inflow of $63,367$ enlisted recruits in FY09 and $69,417$ recruits in FY10 across all MOS. These numbers reflect the recruits after the initial loss during the time in between signing the contract and BT. The total numbers of MOSs in FY09 and FY10 were 152 and 147,

respectively, which is reflective of force structure shifts and MOS changes between years. The table shown in Figure 5.1 details the number of MOS per FY and the total number of recruits.

Table 5.1: Total Recruits and MOSs during FY09 and FY10 by US Army Recruiting Command Data

FY	**Total MOS**	**Total Recruits**
09	152	63,367
10	147	69,417

Of the total number of MOSs, there exists a much smaller set that has a high demand for the formation of BCTs. The Army has addressed some of these high demand, high influx MOSs by creating One Unit Stations or OSUT. For the soldiers of these MOSs, BT and AIT are conducted in the same location with no break in between the two courses. The soldiers of these particular MOS that train in OSUT, surpass the other MOS volumes significantly to make OSUT a feasible option. Therefore, while there are several MOSs possible as shown in Table 5.1, we will narrow our focus on the select few MOS that cause the most scheduling issues for the Army when fulfilling BCT requirements. Some of these MOSs create problems because of the large number of soldiers flooding the training system and also a high demand for fulfilling BCT requirements regularly. Therefore, scheduling of finite resources to accommodate all the soldiers can be tedious. Other MOS issues are opposite to this scenario in the sense that there are so few people of a particular MOS, often called low density MOS, but they are still required to fulfil the BCT requirements. After many discussions with various points of contacts at TRADOC G-3/5/7 and Human Resource Command (most recent discussion took place in February 2011), we converged to the MOSs listed in Table 5.2.

For just these selected MOSs, a total of 15,710 soldiers entered the training system during FY09 and FY10. Of the list of MOS in Table 5.2, the two MOS that have the highest volume during FY09 and FY10 are the 88M and 91B soldiers. Since recruiting targets are

Table 5.2: Selected MOS with Description (source: Army-Portal)

MOS	**Description**
25U	Signal Support Systems Specialist
35M	Human Intelligence Collector
68W	Health Care Specialist
88M	Motor Transport Operator
91B	Wheeled Vehicle Mechanic
92A	Automated Logistical Specialist

partially based on the force structure of units, these volumes reflect the BCT requirements of these particular MOSs. The next two MOSs with the highest volume are the 25U and 68W soldiers. Lastly, the MOSs 92A and 35M have the least number of soldiers. For some MOSs, such as 91B, the MOS was changed from 63B to 91B. Therefore, the values in FY09 from October or July are the values of the 63B recruits. The course resources during AIT for these two MOSs are the same. The monthly recruits for these MOSs are listed in the charts displayed in Tables 5.3 and 5.4. These monthly totals of recruits reveal requirements on Basic Training resources. Recalling that all the MOSs train together during BT, these sum totals across MOS reflect the monthly volumes during FY09 and FY10 for these six MOSs.

Table 5.3: FY09 monthly recruit numbers for the selected MOS (source: US Army Recruiting Command)

MOS	Oct	Nov	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep
25U1	185	138	235	90	148	89	61	198	192	173	163
35M1	0	91	165	0	71	127	0	94	0	112	96
68W1	219	380	346	326	403	255	181	474	322	190	475
88M1	326	202	469	242	186	145	38	172	87	225	207
91B1	218	285	391	232	218	120	63	258	210	303	267
92A1	161	147	189	151	84	94	88	184	65	239	170

As discussed previously, during AIT, the soldiers separate according to their MOS and train in the particular skill sets necessary for their respective MOS. The training durations

Table 5.4: FY10 monthly recruit numbers for the selected MOS (source: US Army Recruiting Command)

MOS	Oct	Nov	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep
25U1	127	141	109	55	31	71	6	63	74	91	104
35M1	0	143	169	1	146	142	2	120	0	83	86
68W1	512	165	565	285	357	299	394	322	284	405	250
88M1	200	298	257	241	217	251	270	352	293	200	125
91B1	249	260	244	129	160	199	138	221	216	268	266
92A1	99	145	165	151	167	134	140	291	225	115	126

of these different MOSs are listed in Table 5.5. We define the term "Instructor," mentioned in the second column of Table 5.5, as one class session. In actuality, there maybe multiple instructors for a particular course being offered; however, in this reference to the number of instructors, it is the ability to offer multiple sessions of classes simultaneously. Compared to the number of people in each MOS shown in Tables 5.3 and 5.4, the number of instructors (or class size) reflects the recruits for respective MOSs volumes. The 88M and 68W MOSs have large class size ranges and a moderate number of instructors. The 25U, 35M, 91B and 92A recruits have smaller class sizes but a greater number of instructors in comparison.

Table 5.5: FY09-FY10 Advanced Individual Training Data (source: Army Training Requirements and Resources System)

MOS	**Number of Instructors**	**Processing Time** (months)	**Min**	**Max**
25U	27	3	18	32
35M	18	4.6	14	36
68W	6	3	120	450
88M	7	1.75	100	120
91B	15	3	30	47
92A	32	2.3	25	30

After AIT, the soldiers of the various MOS assemble into a unit fulfilling several of the skill-level requirements. The data available for determining beginning windows for the formation of BCTs' soldier assignment conforms with the formation of a division. The chart

in Figure 5.1 graphically details the makeup of a division as it is included in a Field Army. Using the Force Management System Website, all the BCTs were broken out in an effort to find their specific manning requirements. The timeframe for the manning window was calculated based on the "reset" start time of the division. The beginning window starts during the month the division starts "reset" and ends three months after the start time. Typically, in ARFORGEN, there is a six-month window to fulfil all requirements; however, newly graduated IMT soldiers are preferred to arrive to the unit earlier. The list of all the divisions following ARFORGEN from FY09 to FY10 with their beginning window (BW) and calculated ending window (EW) are shown in Table 5.5.

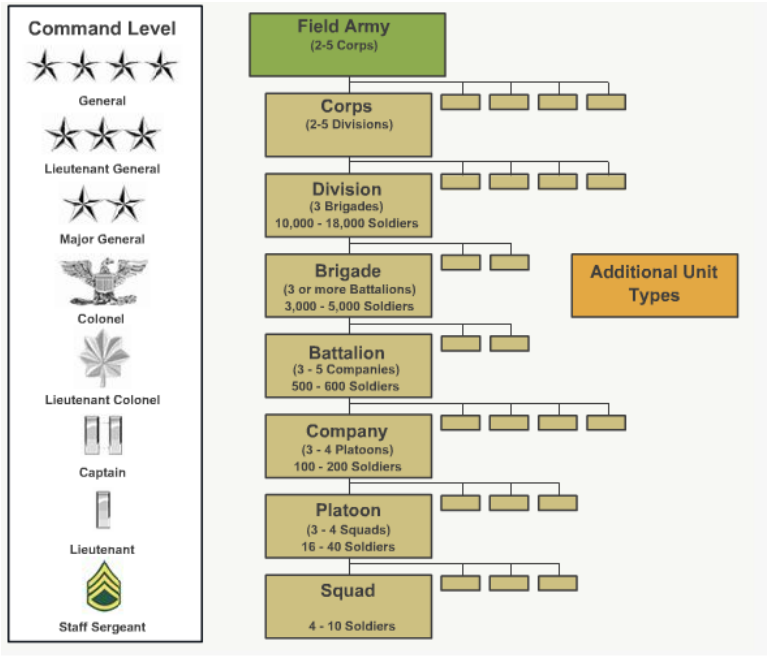


Figure 5.1: Army Operational Unit Diagram (source: US Army)

Table 5.6: FY09-FY10 Brigade Combat Team Schedule (source: Human Resource Command G-3)

Unit	**BW**	**EW**
1I2 - 2/1 ID (HBCT)	Oct-09	Dec-09
1I3 - 3/1 ID (IBCT)	Jul-09	Oct-09
41I - 4/1 ID (IBCT)	Aug-10	Nov-10
11A - 1/1 AD (HBCT)	Aug-10	Nov-10
21A - 2/1 AD (HBCT)	Jun-09	Sep-09
1A4 - 4/1 AD (HBCT)	May-10	Aug-10
11C - 1/1 CD (HBCT)	Feb-10	May-10
21C - 2/1 CD (HBCT)	Jan-10	Apr-10
31C - 3/1 CD (HBCT)	Dec-09	Mar-10
1C4 - 4/1 CD (HBCT)	Jun-09	Sep-09
2I2 - 2/2 ID (SBCT)	Jul-10	Oct-10
32I - 3/2IN (SBCT)	Aug-10	Nov-10
42I - 4/2 ID (SBCT)	Sep-10	Dec-10
43I - 4/3 ID (IBCT)	Dec-08	Mar-09
14I - 1/4 ID (HBCT)	Mar-09	Jun-09
24I - 2/4 ID	Sep-09	Dec-09
34I - 3/4 ID (HBCT)	Feb-09	May-09
4I4 - 4/4 ID	Jun-09	Sep-09
11M - 1/10 MTN (IBCT)	Nov-08	Feb-09
21M - 2/10 MTN (IBCT)	Aug-08	Sep-08
31M - 3/10 MTN (IBCT)	Jan-10	Apr-10
41M - 4/10 MTN (IBCT)	Jan-09	Apr-09
251 - 1/25 ID (SBCT)	Sep-09	Dec-09
225 - 2/25ID (SBCT)	Mar-09	Jun-09
325 - 3/25ID (IBCT)	Oct-09	Jan-10
425 - 4/25 ID (IBCT)	Feb-10	May-10
182 - 1/82 ABN (IBCT)	Aug-10	Nov-10
382 - 3/82 ABN (IBCT)	Dec-09	Mar-10
482 - 4/82 ABN (IBCT)	Aug-10	Nov-10
101 - 1/101 AA (IBCT)	Dec-08	Mar-09
201 - 2/101 AA (IBCT)	Nov-08	Feb-09
301 - 3/101 AA (IBCT)	Nov-08	Feb-09
401 - 4/101 AA (IBCT)	Mar-09	Jun-09
170 - 170th IN (AOE)	Jun-09	Sep-09
172 - 172nd IN (AOE)	Nov-09	Feb-10

5.2 Performance Comparison of Solution Methodologies

We have presented three different solution methods for solving the AFGP. In Chapter 3, we formulated the problem as a lot streaming problem and obtained the solution by CPLEX. Only small-sized problems can be solved to optimality with this method before the system runs out of memory. In an effort to improve this solution approach, in Chapter 4, we presented a column generation-based solution method that requires reformulation of the problem. The reformulation enables decomposition of the problem into subproblems such that a master problem chooses the best schedule from two subproblems that generate different parts of a larger schedule. This problem is now referred to as the "Combined" problem since it combines the first (BT) and second (AIT) stages of the system into one subproblem. Our investigation to improve this method led to an important result that allows us to break up the first part of the Combined reformulation so that the first and the second stages of the system can be further decomposed and solved sequentially such that the completion times obtained at the BT stage are passed and used as a bound on the completion times for the AIT stage. We designated this method as the "Sequential" method.

To compare the solution methods in solving the AFGP, we fixed some parameters of problems for each of the solution methods. In Table 5.7, all the timing parameters and the number of ARFORGEN units are listed. The unit of timing parameters is months. Note that the parameters that vary with skills are fixed to be the same across all the skills and therefore listed only once. The biggest problem tested on all three models has 20 soldiers. The ready times for soldiers were kept consistent across models and are listed in Table 5.8. The unit of ready times is also months, that is for the first soldier arriving at time 0.5, he arrived in the first half of the first month. The common due window times for the BCTs and Army units are listed in Table 5.9 for each unit.

Recall that we have referred to the decomposition of the problem achieved using

Table 5.7: Parameter values for sample problems

Parameters	Values
$B^{(1)}$	4
$P^{(1)}$	2
D	2
$B^{(2)}$	4
$I^{(2)}$	3
$L^{(2)}$	1
$U^{(2)}$	5
Total Units	5
ARFORGEN	3

the combined method as Decomposition I and the one based on the sequential method as Decomposition II. The results obtained using the three methods are shown in Table 5.10. As shown previously in Table 3.3, the results for the direct solution by CPLEX are few since it runs out of memory at any point the model starts getting larger. One of the observations from this chart and as discussed in Chapter 3, is that as the number of MOS increases for a fixed number of soldiers, the problem becomes easier to solve.

The CPU times for Decomposition I, also shown in Table 5.10, are sometimes less than and sometimes more than that required by CPLEX. Decomposition I runs were stopped around four hours instead of being allowed to continue. In some cases, the only benefit of the Combined model over the CPLEX model is that the Combined model would run, albeit for hours to completion instead of reaching the out of memory status. While the CPLEX model is overly complex, the Combined model decomposed and solved the problem via several enumerations per iteration. One disadvantage of this Combined solution approach is that it does not experience the same time savings as the CPLEX model experiences with the increase in MOS. Therefore, the Combined problem seems to do better than the CPLEX model for fewer MOS but more soldiers.

Decomposition II method improves on both direct solution by CPLEX and Decom-

Table 5.8: Ready Time values (in months) for each soldier

Soldier	Ready Times
1	0.5
2	0.5
3	1
4	1
5	1
6	1
7	1.5
8	1.5
9	1.75
10	1.75
11	2
12	2
13	2.25
14	2.25
15	2.5
16	2.5
17	2.75
18	2.75
19	3
20	3

position I by being able to solve larger models and solve them faster. By being able to break out the first two subproblems in the Combined problem, the Sequential method displays the advantage of the decomposition approach across the number of MOSs in the second stage by being able to generate AIT schedules for each MOS as separate problems given the BT completion times. Therefore, as the number of MOSs increase, the Decomposition II method experiences some of the same time savings as direct solution by CPLEX. Each of the models was run on an E3535 2 Ghz 3.25G RAM quad core or equivalent using CPLEX 12.1, OPL 6.3.

Based on our preliminary results shown in Table 5.10, we perform more extensive testing to understand the behaviors of the three different solution methods. For different variations of the AFGP model, we perform ten runs for different sets of unequal ready times.

Table 5.9: Sample BCT and Army Unit demand windows

Unit	Beginning Window	Ending Window
BCT 1	5	10
BCT 2	5	10
BCT 3	7	10
Army 1	9	12.5
Army 2	10	13

Table 5.10: CPU times of different solution methods (in seconds)

Number of MOS	Number of Soldiers	CPLEX	Decomposition I	Decomposition II
2	5	4.39	5.53	2.73
2	10	1,496.34	106.28	2.29
2	15	out of memory	out of memory	3.55
2	20	out of memory	out of memory	629.23
3	5	2.90	8.02	3.00
3	10	91.48	152.69	3.63
3	15	out of memory	out of memory	13.45
3	20	out of memory	out of memory	1,085.38
4	5	2.71	520.89	2.51
4	10	28.68	65.78	2.51
4	15	out of memory	$> 14,400$	6.23
4	20	out of memory	$> 14,400$	851.61
5	5	2.71	$> 14,400$	2.92
5	10	4.29	2,089.38	2.97
5	15	193.27	$> 14,400$	5.59
5	20	out of memory	$> 14,400$	213.27

Model variations include changes in the following parameters: (i) the number of MOS, (ii) number of instructors, and (iii) the number of soldiers. Most of the parameter settings used in the preliminary model were kept the same. However, we list the parameters that were varied in Table 5.9 with the exception of the changes in beginning and ending of windows. We fix all beginning and ending windows to five and ten, respectively, to cover the range of our unequal ready time sets. For these test problems, we use a different computer to handle larger problem instances. The computer is equipped with an AMD Phenom II 3.3GHz six-core desktop processor with 16GB SDRAM DDR3 using CPLEX 12.2.

Table 5.11: Parameter value changes from previous settings

Parameters	Values
$B^{(1)}$	3
$B^{(2)}$	3
$L^{(2)}$	2

Given the limitations of Decomposition I CPLEX, we divide our testing into three categories: (i) small-sized problem instances, (ii) medium-sized problem instances, and (iii) large-sized problem instances. For the small-sized problem instances, we compare all three solution approaches. We also show the accuracy of the solutions obtained using the Decomposition II method by the percent gap in its solution value and that of the direct solution obtained by CPLEX. The averages of ten different ready time runs for each model variation for small-sized problem instances are shown in the graph displayed in Figure 5.2. For the medium-sized problem instances, we compare solution values obtained using Decomposition II with those obtained directly using CPLEX. We also present the accuracy of Decomposition II by computing the average percent gap between the values of solutions obtained by Decomposition II and optimal solution. The average of the ten different ready time runs for each model variation for medium-sized problem instances are shown graphically in Figure 5.3. In the case of large-sized problem instances, only Decomposition II is tested. The large-sized problem instances are too large to be solved directly by CPLEX. We therefore, only report the CPU times required by Decomposition II to understand its behavior for large-sized problems. The average of the ten different ready time runs for each model variation for large-sized problem instances are shown graphically in Figure 5.4.

Observing the performances of the different solution methods in the graph shown in Figure 5.2 for small-sized instances, note the the relatively equal performances of Decomposition II and CPLEX. For small-sized instances, the Decomposition II method requires the same amount of CPU time as that required for solving the problem directly using CPLEX.

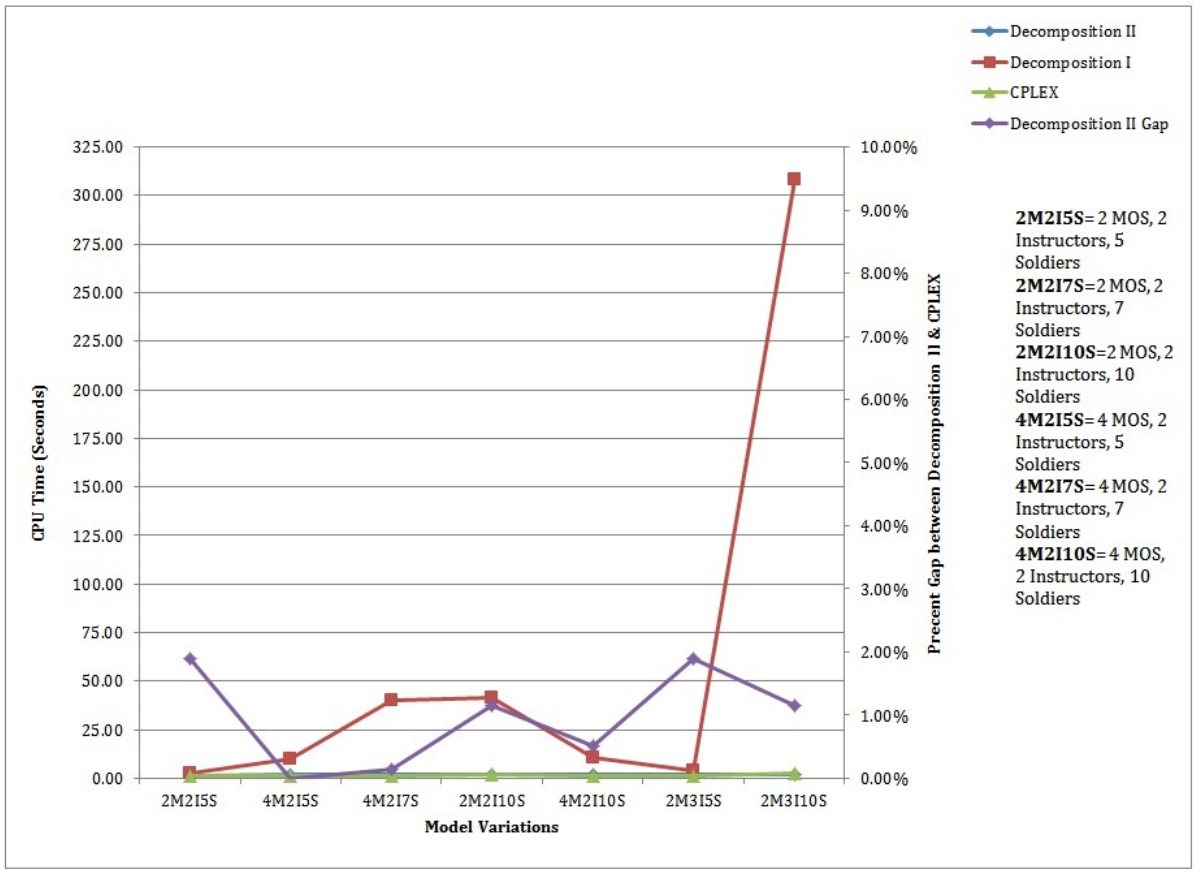


Figure 5.2: AFGP Averages per Small-Sized Model Variation Runs for Decomposition I, II and CPLEX, and also, Optimality Gap between Decomposition II and CPLEX

Decomposition I, however, consistently requires more CPU time than both Decomposition II and direct solution by CPLEX. The only model variations for which Decomposition I comes close to the performances of Decomposition II and direct solution by CPLEX are for very small-sized instances of 5 soldiers. In Figure 5.2, we also observe the accuracy of Decomposition II which when compared with the optimal solution as shown in the graph (please see the right vertical axis for optimality gap). For all model variations, the gap percentage is always less than 2%, and for the majority of cases the gap is less than 1%.

The average performances of Decomposition II and solutions obtained directly using CPLEX are shown in Figure 5.3 for medium-sized problem instances. We observe similar CPU times required by the two approaches for (the smaller) 15-soldier scenarios, with a

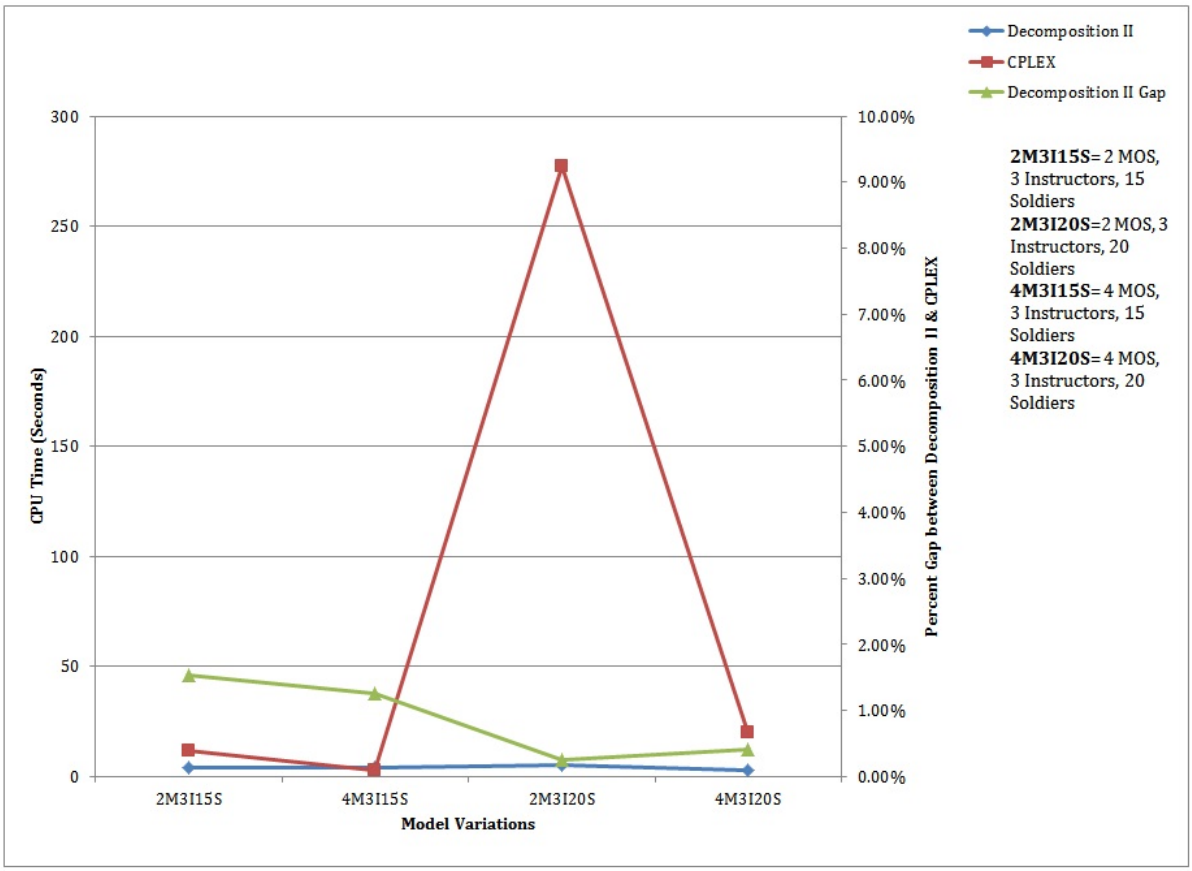


Figure 5.3: AFGP Averages per Medium-Sized Model Variation Runs for Decomposition II and CPLEX, and also, Optimality Gap between Decomposition II and CPLEX

reduction in CPU times as the number of MOSs is increased. When the size of the problem is increased to 20 soldiers for the 2 MOS case, the time required by the solution obtained using CPLEX significantly increases. In contrast, the average CPU time required of Decomposition II is about the same as for the previous 15-soldier case. In the last model variation with 4 MOS and 20 soldiers, the average required CPU time reduces significantly. Previously, we observed the solutions obtained directly by CPLEX to require less CPU time as the number of MOSs increases. We note also the slight reduction in CPU time required by Decomposition II for the same case. The optimality gap time obtained by Decomposition II as shown in Figure 5.3, is always less than 1.60%. Note a reduction in optimality gap for two larger cases with 20 soldiers ($2M3I20S$ and $4M3I20S$).

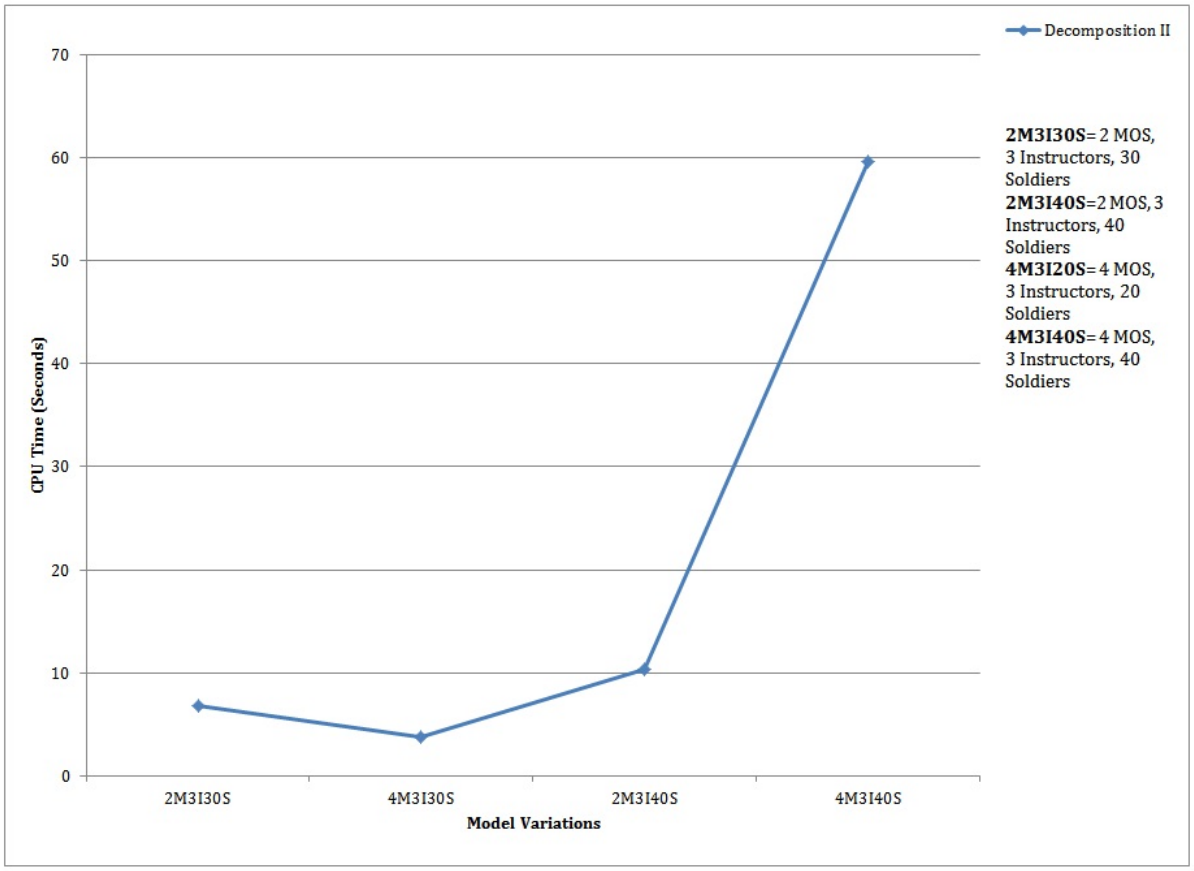


Figure 5.4: AFGP Averages per Large-Sized Model Variation Runs for Decomposition II

In Figure 5.4, the average performance of Decomposition II for large-sized instances is shown. Overall, Decomposition II solves larger problems in about ten seconds or less. The one run with 4 MOS and 40 soldiers had two runs that took several minutes compared to the rest. Nevertheless, even 40 soldiers for 4 MOS took on average less than a minute to solve.

5.3 Recommendation and Discussion of Brigade Combat Team Problem

As discussed earlier in Section 5.1, the AFGP problem clearly can be very large, even for six military occupational specialties. Most of the data presented in Section 5.1 covers a two-year time frame, which is an unreasonable length of time to consider for a planning horizon. Hence, selecting the appropriate planning horizon of the AFGP problem will reasonably allow us to solve portions of the overall problem by solving for some fixed planning horizon. The appropriate planning horizon is based on the size of the BCT requirement windows. The full planning horizon spans the period from the time the recruits enter the system to the time they are ready to form the BCT units. By using a full due window (three months) as a planning target, we must calculate backwards to find the relative recruits for each MOS (using historical data) to meet their demand. As a result of using the three-month window, our planning horizon spans a year.

After determining the duration of the planning horizon, the data requirements are calculated essentially backwards. That is, for a given planning horizon, we calculate the number of BCTs that enter the reset phase and their requirements as well as those of the remaining Army requirements. Using a range of flow time respective of each MOS, the number of recruits added to the model are calculated by determining the number of months before and during the units' reset phase is feasible. For example, for a three-month planning horizon, we determine the number of BCTs and their requirements during that time. In order to calculate the number of recruits to add to the system, we determine the minimum and maximum flow time of a particular MOS. Continuing with the example, consider a minimum and maximum flow time of eight and twelve months, respectively. The number of recruits that would be considered are all the recruits coming in during the months falling in the range of the beginning of the planning horizon minus twelve months and the ending of the planning horizon minus eight months. All the recruits that enter the training system in

this range capture the earliest and latest times the recruits could enter the system and be assigned to any of the units within the planning horizon.

To reduce the problem to a more manageable size, we investigate the idea of aggregating soldiers with the same ready times and MOS. Using the planning horizon and this idea of aggregation, the most logical approach is to solve an aggregated three-month BCT window problem. The aggregation is slightly more approximate than solving for example, one-month BCT windows; however, linking the multiple one-month BCT demand windows is more difficult. In the one-month BCT demand window planning horizon, some instructors for different MOS will still be utilized by the previous planning horizon, thus making class schedules harder to link over multiple periods. Also, BCTs for the one-month demand window will only be partially fulfilled and must be linked for every BCT and rest of Amy demand. Since the BCT demand window is of three-month duration, solving for a more aggregated solution still generates a good approximation (based on the results). Solving the three-month window problem fulfills a set of units during each period of the planning horizon. To solve for a year of BCTs entering the system, only four problems would have to be solved. The linking of the planning horizons is less complicated compared to the one-month problem. In the three-month case most all instructors will be available by the time the next set of recruits start AIT. To account for the greater aggregation, sometimes a couple of instructors must be used to accommodate the aggregation ratio of soldiers while maintaining the class size restrictions. Another possible approach is to aggregate for a six-month planning horizon. For this case, we would only have to solve the model twice and link two planning horizons. The trade-off of this approach is the greater loss of data fidelity as we aggregate larger quantities of soldiers. The class sizes and number of instructors will loose the ability to have multiple sessions that start at different times if to form one class would require several instructors.

We investigate both the one-month and three-month BCT window cases to solve AFGP to compare the approximate solutions. In Tables 5.12 and 5.13, the total number

of recruits used for each case are presented along with the ratio value we use to aggregated soldiers. For example, in Table 5.12 every one soldier for the 25U represents 30 soldiers of that same MOS. The next column "Aggregation Totals" gives the total number of soldiers we truly represent through the aggregation. For the one-month BCT window case, the total difference in soldiers is only seven. For this particular model, even though the target is to fulfil one-month of the BCT demand window, the planning horizon for this case is 11 months, from the time recruits enter the system to the due window month.

Table 5.12: One-Month BCT Due Window Total Recruit Numbers: Aggregated vs. Actual Soldier Totals

MOS	Ratio	Aggregation Totals	Actual Value
25U	30	390	390
35M	16	128	127
68W	79	790	796
88M	45	315	312
91B	120	480	468
92A	41	246	249
Total Ratio vs. Actual Difference: 7			

Table 5.13: Three-Month BCT Window Total Recruit Numbers: Aggregated vs. Actual Soldier Totals

MOS	Ratio	Aggregation Totals	Actual Value
25U	200	800	726
35M	50	300	292
68W	120	1440	1461
88M	100	700	719
91B	120	1080	1038
92A	100	600	658
Total Ratio vs. Actual Difference: 26			

With both the one-month and three-month BCT window target models, the aggregation of all the soldiers changes the parameter values that we must account for in the model. In Tables 5.14 and 5.15, we present new parameters relative to the ratios of soldiers for each

MOS. The class sizes and the number of instructors most adjust with the number of soldiers now represented by a single soldier. The processing times remain unchanged.

Table 5.14: One-Month BCT Window: AIT Parameters

MOS	Number of Instructors	Processing Time months	Min	Max
25U	3	3	1	3
35M	3	4.6	1	2
68W	3	3	2	5
88M	3	1.75	2	3
91B	3	3	1	2
92A	3	2.3	1	2

Table 5.15: Three-Month BCT Window: AIT Parameters

MOS	Number of Instructors	Processing Time months	Min	Max
25U	3	3	1	1
35M	3	4.6	1	2
68W	3	3	1	4
88M	3	1.75	1	2
91B	3	3	1	3
92A	3	2.3	1	1

The total number of soldiers required for each BCT and Army units must also be adjusted to account for the aggregated representation of soldiers. The requirements for both the one-month and three-month BCT windows are shown in Tables 5.16 and 5.17. While the three-month BCT window requirements shown in Table 5.17 look to be fewer than the one-month BCT window requirements shown in Table 5.16, the aggregation ratios in the three-month BCT window case are greater than those for the one-month case.

We compare our approximate solution using a three-month BCT window with that for the more detailed one-month BCT window. Though the total actual number of soldiers in the three-month approximation is far greater than that in the one-month window case, the aggregated version of the three-month model is slightly smaller than the aggregated one-month model. Observing the results in Table 5.18, the mean flow time (MFT) per soldier

Table 5.16: One-Month BCT Window: ARFORGEN and Army Unit Parameters

	MOS					
	25U	35M	68W	88M	91B	92A
BCT	1	1	1	2	1	1
Army	12	7	9	5	3	5

Table 5.17: Three-Month BCT Window: ARFORGEN and Army Unit Parameters

	MOS					
	25U	35M	68W	88M	91B	92A
BCT	1	1	4	5	6	2
Army	3	5	8	2	3	4

of the two are very close. The results essentially differ only by days for each measure as the results are shown in months. We also found no soldier to be tardy for either the ARFORGEN case or the rest of the Army demand for the three-month case. From Table 5.18, the average time a soldier waited for either BT or AIT combined was less than a month (IMT only Mean Flow Time). Soldiers were also aligned with the units as they were on average early only by about a month.

Table 5.18: One-Month vs. Three-Month BCT Window: Performance Measure Results (in months)

Measure	1-Month	3-Month
Mean Flow Time	6.965	6.180
Mean Earliness & Tardiness	1.058	0.810
IMT only Mean Flow Time	5.907	5.378
Mean Wait Time	0.910	0.455

From actual data we received from TRADOC G-3/5/7, on average, soldiers wait at least two months after completing BT. Depending on the MOS, soldiers may wait even longer. For example, for the 35M MOS, soldiers waited on average five-months before starting AIT, in addition to the average two months everyone waited. Given these average wait

times observed for the actual scenario we calculated the actual data estimate relative to our scenario. We calculate the actual data IMT mean flow time estimate for our scenario to be 7.469 months. Compared to our 3-month mean flow time for IMT of 6.189 months, our solution to the AFGP results in schedules with an estimated 28% reduction in mean flow time for soldiers in IMT over what is currently experienced in practice. In practice, new graduates are used only to fill in gaps because their courses do not align with units following ARFORGEN as is. The target duration for ensuring fulfillment of a unit is given to be six months (the whole duration of Reset Phase) and a small period of time in the next phase of ARFORGEN (Ready Phase). We use much stricter due windows of three months as our targets to meet the unit demand for skill level 10 soldiers of a particular set of MOS. All of the skill level 10 soldiers requirements of the BCT were met on time given our more restrictive windows. Our model therefore aligns new soldier recruits to exit IMT in such a way that the soldier with the right skills and quantity arrive to their unit in enough time to fulfill the unit requirements.

From the gantt chart shown in Figure 5.5, we see our large classes were formed to handle all the recruits at the BT Stage. The charts for Stage 2 for each MOS are presented in partially aggregated form to best present the information. For the 25U shown in the first chart in Figure 5.5 each class represents six concurrent sessions with six instructors. Therefore, each class of 200 soldiers is actually conducted across six concurrent sessions amounting to a total of the 200 soldiers. With the BCT and rest of Army windows of 11 and 14, we see all but one class (and concurrent sessions) of soldiers' completion time from AIT were within the due window. The second AIT chart in Figure 5.5 shows the classes for the 35M. For each class size of 100, there are 3 concurrent sessions being taught by 3 instructors totalling to the number of soldiers shown in the block. From their completion times we see all the the 35M soldiers finish AIT before the beginning window and therefore early. The third AIT chart in Figure 5.5 shows the classes for the 68W soldiers. Since the original class and maximum and minimum are large, each class has only two concurrent sessions. From

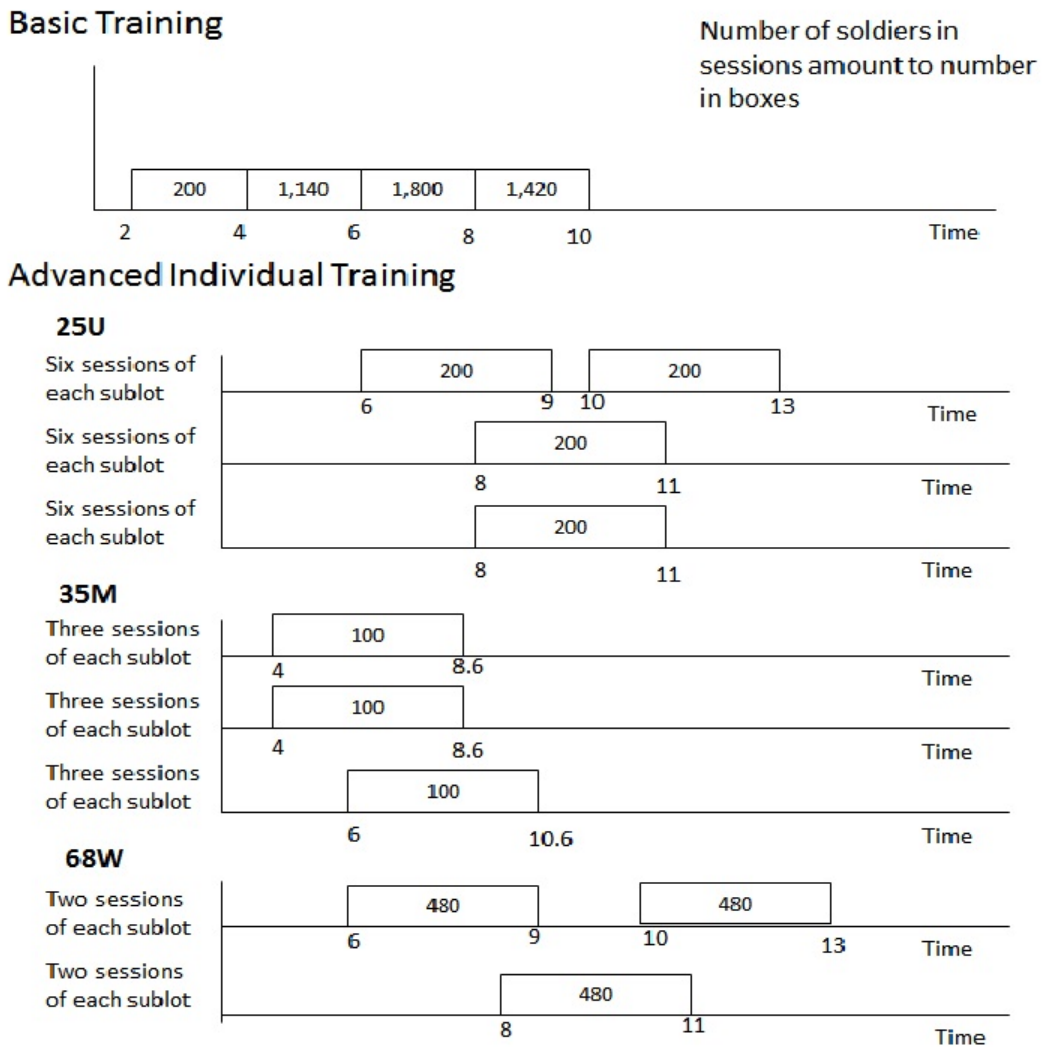


Figure 5.5: Results: Basic Training and AIT Schedules

their completion times, we see only soldiers in the classes finishing at time 9 are early to the units.

The gantt charts for the AIT of each MOS continues in Figure 5.6. The first chart in Figure 5.6 shows the classes for the 88M soldiers. The classes that show 200 soldiers require two concurrent sessions with two instructors. The class size of 100 only needs one instructor. The completion times of 9.75 for two of the classes (and subsequently their concurrent sessions) for the 88M soldiers occur before all the beginning windows of all the units. All the soldiers in these classes incur earliness. The other two classes (and their concurrent sessions)

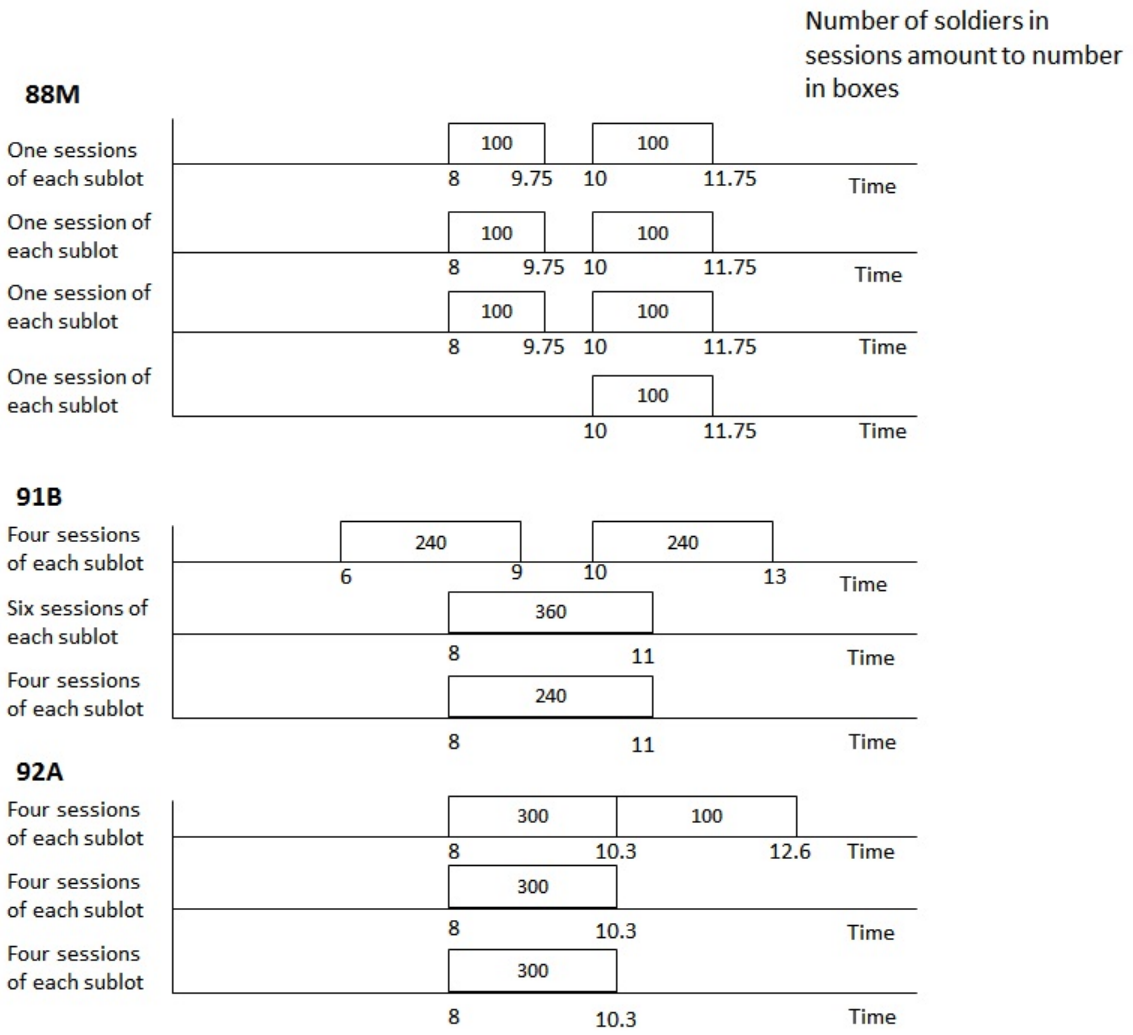


Figure 5.6: Results: AIT Schedules Continued

occur after the beginning windows but before the ending window, thus 88M soldiers in these classes do not incur any penalty. The second chart shows the class schedule for the soldiers of 91B MOS. For the classes of size 240, these classes represent four concurrent sessions using four instructors, totalling to the respective number of their block, all at the times indicated in the chart in Figure 5.6. The class size of 360 represents six concurrent sessions of a class using six instructors at the times indicated. Given the completion times of AIT, all but the first set of soldiers (ending at time 9) fall within requirement windows. The third chart shows the class schedule for the 92A MOS. Each class of 100 represents four concurrent sessions

requiring four instructors. With all the classes and their concurrent sessions ending at time 10.3, except for the one class ending at 12.6, the majority of the 92A soldiers are early.

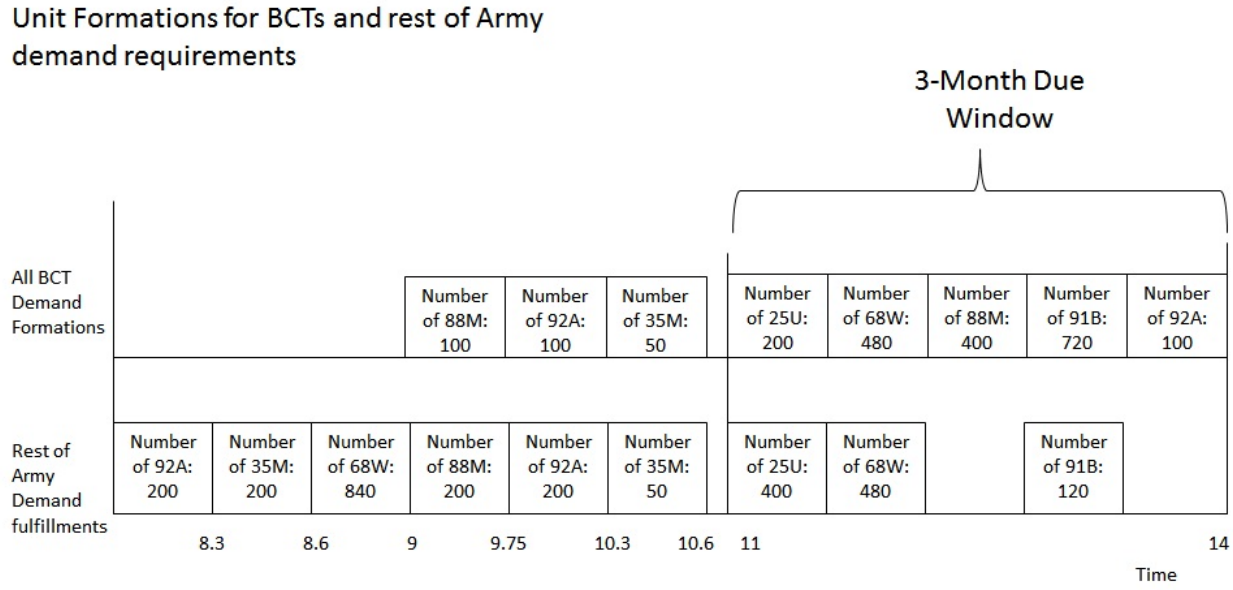


Figure 5.7: Results: Unit Assignments

The chart in Figure 5.7 shows the arrival of soldiers to units in relation to the window. We observe several groups of soldiers arriving shortly before the beginning windows. The earliest any soldiers arrive are those of the 92A MOS that finish AIT at time 8.3. We also observe the majority of the soldiers for the BCTs arrive within the window. All unit requirements are met before the ending window.

To combine this solution to the solution of the next planning horizon another 3-month window would be added to the unit gantt chart showing batches of soldiers being assigned after their completion from AIT. Likewise, additional classes would appear on the gantt chart for each MOS until we had classes (with multiple sessions) over several time periods. The set of recruits would start with the first month not used in the previous planning horizon. Given this case is modeled as un-capacitated new classes for BT can be formed without regard to the previous planning period classes.

While we know from actual data the wait time averages after BT, we cannot quantify

the fulfillment of BCTs used in our example using new graduates. We do know our methodology for aligning schedules to ARFORGEN BCTs demands is different than the current Army processes. Aforementioned, we use stricter BCT windows to fulfill BCTs with their respective MOS requirements. The Army uses twice the length of time to fulfill units. To meet these BCTs requirements the Army currently only uses the new graduates of IMT to close manning gaps of BCTs. Our results of our model indicate the BCTs requirements were completely fulfilled without any soldier being tardy. Unlike our approach the IMT process is not aligned with the demands following ARFORGEN. Therefore, soldiers are scheduled through the two stages of training as quickly as possible without regard to incoming BCTs requirements. The result of this disconnect forces planners to only use the new graduates as a last resort to fill in gaps instead of effectively leveraging them to fulfill BCTs. One of the differences in our approach, is to link the output stream of the new IMT graduates to the BCTs demand requirements following ARFORGEN as well as the rest of the Army demands. By aligning the class completion times to the ARFORGEN windows, the BCT requirements were fulfilled by their due date if not earlier in our example using historical data. Through the alignment of these courses to ARFORGEN, the Army will be better positioned to effectively plan BCT fulfillments using new IMT graduates, as opposed to the new graduates being used as a last resort to fill manning gaps as ad hoc fixes.

5.4 Variations of the Brigade Combat Team Problem

The AFGP discussed is an example of one configuration of the scenario based on the training system and ARFORGEN. For the AFGP, unequal ready times were more appropriate, resulting in a harder model than the case for equal ready times. In the equal ready time case where all jobs arrive at the same time, the problem decomposes more naturally and is one of the only times the Decomposition II method is optimal. For eight different model variations that include changing number of MOS, number of instructors and number of soldiers, we

run ten runs changing only the set of processing times for AIT for each model scenario. In Figure 5.8 we show the CPU times required by Decomposition II and CPLEX.

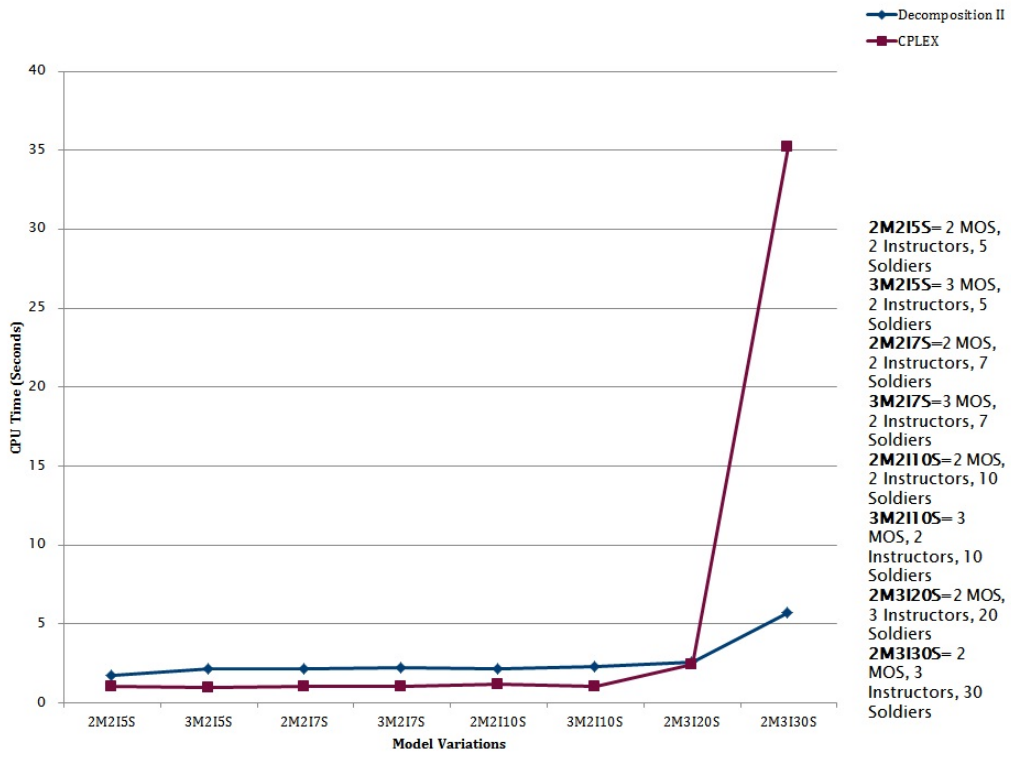


Figure 5.8: Equal Ready Time Averages per Model Variation Runs for Decomposition II and CPLEX

From the graph in Figure 5.8, we observe that for smaller instances of the model variation where there are only five, seven or ten soldiers, Decomposition II and CPLEX require about the same amount of CPU time. As the the number of soldiers increases to 20 soldiers for the *2M3I20S* scenario, the time required by CPLEX to solve the problem begins to increase. When the number of soldiers increases to 30 soldiers for the *2M3I30S* scenario, CPLEX increases significantly as the average number of runs for this case requires more than 35 seconds to solve to optimality. In contrast, Decomposition II increased at a far slower rate from the 20 to 30 soldier model scenarios. For the same scenario that required 35 seconds for CPLEX, Decomposition II only required 5.7 seconds to solve to optimality.

Chapter 6

Extensions of the AFGP

6.1 Hybrid Flow Shop

One of the more common hybrid flow shops discussed in the literature is a flexible flow shop. The flexible flow shop generalizes the flow shop and parallel machine environment by having at least one stage in a flow shop with multiple identical (or parallel) machines [58]. This particular example of a hybrid flow shop has received plenty of attention in the literature. The generalization of the flow shop to include parallel machines makes the flexible flow shop environment complex and thus hard to solve. In a brief review of the literature, mostly heuristic methods have been proposed to solve the flexible flow shop problem.

Brah and Hunsucker [8] investigate minimizing maximum completion time for a flexible flow shop. In their survey of the literature at the time of their publication, they report that the literature provides formulations and solutions that can only solve small size problems. This problem is NP-Hard, and it is a general version of the flow shop problem, which is shown to be NP-Hard. They propose the use of a branch-and-bound method for the solution of medium-sized problems in reasonable times.

Gupta and Tunc [36] approach larger-sized instances of a two-stage hybrid flow shop with parallel machines at the second stage. Specifically, Gupta and Tunc explore a heuristic method along with a branch-and-bound method to solve large-sized problem instances that are computationally extensive. The problem is split into two parts: sequencing of jobs, and assigning of jobs to machines. By performing these steps independently, the solutions generated may not be optimal.

Rajendran and Chaudhuri [61] address a problem similar to that in [36] except that they consider minimizing the total flow time. They employ the use of a heuristic based on a branch-and-bound method. They report results for up to ten jobs and 15 parallel machines at Stage 1.

Gupta and Tunc [37] extend the two-stage hybrid flow shop problem to include separable setups and removal times minimizing the makespan, and also, for the objective of the number of machines used at each stage as a secondary criterion. They propose four heuristic for different setups of the hybrid two-stage problem. These are further extensions of the method presented in [36], first sequencing the jobs, and then, assigning the jobs to the machines independently.They report solutions of large-sized problems in reasonable times and obtain solutions that reasonably close to the optimal solution.

Wang [74] presents a literature review of the work reported on flexible flow shop through 2005. This survey covers about three decades of work on flexible flow shop problems. Of the optimum seeking approaches surveyed, some give only the formulation of the problem for different configurations of the flexible flow shop and solve with a CPLEX solver. Other optimum approaches use branch-and-bound techniques to solve slightly bigger problem instances than those solved using CPLEX. Usually the improvement of these kinds of approaches is achieved through the use of better lower bounds.

Jungwattanakit *et al.* [43] present various established algorithms to solve flexible

flow shop problems when the parallel machines are not identical, there are setup times and multi-objective performance measures. A mathematical model of the problem is presented and solved for optimality using CPLEX for small-sized instances. They also present various heuristic methods to solve large-sized instances of the problem to near optimality. They also compare the constructive algorithms used commonly in scheduling literature to sequence the jobs and then assign them to the first available machine [43]. With the genetic algorithms to solve the flexible flow shop problem with unrelated parallel machines and setup times, they report particular forms of the constructive and genetic algorithms that work best for their particular problem.

Note that the first and second stages of the AFGP can be considered as a hybrid flow shop. In the first stage of the AFGP Problem, multiple jobs are being processed by the same, single machine in batches. As an extension of the AFGP, the first stage can be assumed to consist of m parallel machines. When the jobs transition to the second stage, similar jobs are processed together. There are several parallel machines available at Stage 2 to process jobs of the same type. Unlike the traditional flexible flow shop, the separation of the jobs by type decomposes the problem. A common approach proposed in the literature for this problem is to first sequence the jobs, and then, assign these to machines. However, the structure of our problem enables an effective use of the column generation method that assumes attainment of optimal solutions.

The complexity of the model varies based on the variety of features that may be present namely, ready times, sublot sizes, objective functions, and capacity of machines. A classification of the hybrid flow shop problem is presented in Figure 6.1. The two categories of problems presented in Figure 6.1 are un-capacitated, and capacitated first stage problems. Within the two different categories the problem can break into additional types of machine schedule problems by varying the different features aforementioned such as equal and unequal ready times. Each set of problems are relevant through application examples. For example,

in the AFGP problem unequal ready times for soldiers were considered since in application soldiers arrive at different times throughout a planning horizon. For each of these different model attributes, the objective can also vary and the different objectives are presented in Figure 6.1 for each set of problem scenario in the chart. All the problem scenarios in Figure 6.1 are for a two stage hybrid flow shop where there are multiple parallel machines at both stages. With the different types of problem attributes the complexity of the problem will also vary. For example, when jobs arrive at the same time, the model is less complex than the case where each component arrives at different times. In the solution method presented in Chapter 4, the completion times for each individual soldier is passed to the second stage as a bound on what the completion time for the second stage can be. With all the individuals arriving at distinct times in the first stage, there are a number of possible completion times the first stage could take for a batch of individuals grouped into one class. If each individual arrived at the same time and thus had equal ready times, the bound on the second stage completion time could be based only on parameters of the model, thereby leading to an efficient decomposition method.

In lot streaming problems, scenarios with variable sublots are quite complex as discussed in [7]. With the sublots (or classes as referenced in the problem described in Chapter 3) reforming at each stage, the flexibility adds to the complexity of the problem. In the AFGP, the sublots (referenced as classes) are given a range of their sizes in the second stage. When the sublots are forced to be equal or even consistent, the model becomes less complex. The capacity of each of the m machines for these lot streaming problems is defined as the size of a batch a machine can handle at one time. One of the key aspects of our Sequential approach is the first stage's ability to make a sublot of any size when it is un-capacitated. If this first stage is made capacitated by putting restrictions on the size of the batch a machine can handle, the job's completion times in the first stage could not be passed to the second stage. Therefore, the problem could not be broken into two subproblems to achieve an effective decomposition.

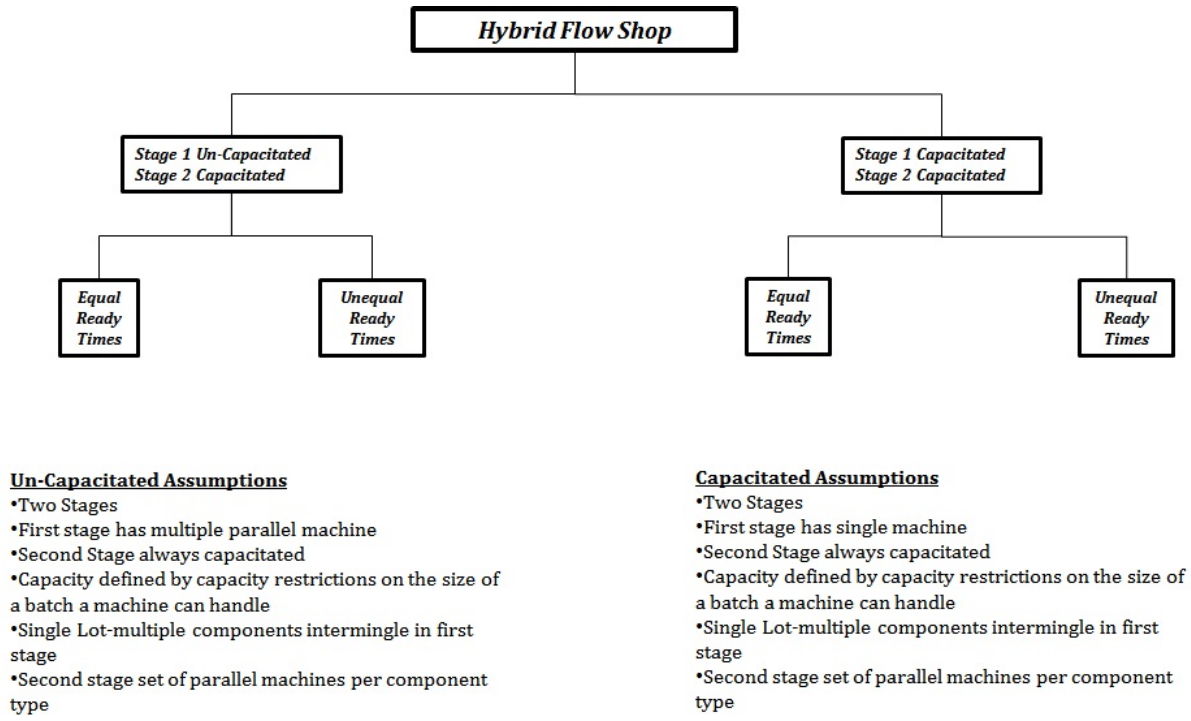


Figure 6.1: Hybrid Flow Shop Extensions

All these possible problem features for our particular hybrid flow shop can be summarized in the chart shown in Figure 6.1.

The following subsections present model formulations of the problem scenarios shown in Figure 6.1. The notations for both the un-capacitated and capacitated problems are almost identical. Next we present the notation for the hybrid flow shop problems for both the master problem and pricing problems.

Notation for Decomposition II Master Problem

Parameters

S	Total number of jobs
σ	Total number of product types
s_l	Number of jobs of product l, $\forall\ l = 1, ..., \sigma$
R_s	Ready time of job s
$P_l^{(2)}$	Processing time per job type for Stage 2
$C_{st}^{(1)}$	Completion time of job s in iteration t an stage 1
$C_{st}^{(2)}$	Completion time of job s in iteration t an stage 2

Variables

$W_s^{(2)}$ Waiting time of job s at Stage 2 before starting its processing, $\forall\ s = 1, ..., S$

$$x_t^{(1)} = \begin{cases} 1, & \text{if schedule } t \text{ is selected at Stage 1,} \\ 0, & \text{otherwise.} \end{cases}$$

$$x_t^{(2)} = \begin{cases} 1, & \text{if schedule } t \text{ is selected at Stage 2,} \\ 0, & \text{otherwise.} \end{cases}$$

Notation for Pricing Problems

Parameters

$P^{(1)}$ Processing time for Stage 1
$B^{(1)}$ Maximum number of sublots per machine for Stage 1
$U^{(1)}$ Maximum sublot size at Stage 1
$L^{(1)}$ Minimum sublot size at Stage 1
$M^{(1)}$ Number of machines at Stage 1
$U^{(2)}$ Maximum sublot size at Stage 2
$L^{(2)}$ Minimum sublot size at Stage 2
$M_l^{(2)}$ Number of machines per job type at Stage 2
$B^{(2)}$ Maximum number of sublots per machine at Stage 2
$\gamma^{(1)}$ Dual variable corresponding to convexity constraint of Stage 1
$\gamma_l^{(2)}$ Dual variable corresponding to convexity constraint of Stage 2
$\mu_s^{(1)}$ Dual variable corresponding to complexity constraint of Stages 1 and 2

Variables

$C_{smi}^{(1)}$ Completion time of job s contained in sublot i on machine a
$C_{slak}^{(2)}$ Completion time job s contained in sublot k on machine a
$\forall\ l = 1, ..., \sigma,\ a = 1, ..., M_l^{(2)},\ k = 1, ..., B^{(2)}$.

$$y_{smi}^{(1)} = \begin{cases} 1, & \text{if job } s \text{ of type } l \text{ is contained in sublot } i \text{ on machine } a \\ 0, & \text{otherwise, } \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., j. \end{cases}$$

$$y_{slak}^{(2)} = \begin{cases} 1, & \text{if job } s \text{ of type } l \text{ is contained in sublot } k \text{ on machine } a \\ 0, & \text{otherwise, } \forall\ s \in s_l,\ a = 1, ..., M_l^{(2)}\ l = 1, ..., \sigma, \\ & k = 1, ..., B^{(2)}. \end{cases}$$

$$X_{im}^{(1)} = \begin{cases} 1, & \text{if sublot } i \text{ on machine } m \text{ is formed} \\ 0, & \text{otherwise.} \end{cases}$$

$$X_{lka}^{(2)} = \begin{cases} 1, & \text{if for job type } l \text{ sublot } k \text{ is formed on machine } a \\ 0, & \text{otherwise.} \end{cases}$$

Notation for MIP Formulation

Variables

$W_s^{(1)}$ Waiting time of job s at Stage 1 before starting processing, $\forall\ s = 1, ..., S$.
$W_s^{(2)}$ Waiting time of job s at Stage 2 before starting processing, $\forall\ s = 1, ..., S$.
$S_{im}^{(1)}$ Size of sublot i on machine m, $\forall\ i = 1, .., B^{(1)},\ m = 1, ..., M^{(1)}$.
$S_{lka}^{(2)}$ Size of product type l of sublot k on machine a, $\forall\ l = 1, ..., \sigma,\ k = 1, .., B^{(2)}$, $a = 1, ..., M^{(2)}$.
$C_{im}^{(1)}$ Completion time of sublot i on machine m, $\forall\ i = 1, ..., B^{(1)},\ m = 1, ..., M^{(1)}$.
$C_{lka}^{(2)}$ Completion time of sublot k on machine a, $\forall\ l = 1, ..., \sigma,\ a = 1, ..., M_l^{(2)}$, $k = 1, ..., B^{(2)}$.

$$Z_{smi}^{(1)} = \begin{cases} 1, & \text{if job } s \text{ of type } l \text{ contained in sublot } i \text{ on machine } a \\ 0, & \text{otherwise, } \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., j \end{cases}$$

$$Z_{slak}^{(2)} = \begin{cases} 1, & \text{if job } s \text{ of type } l \text{ is contained in sublot } k \text{ on machine } a \\ 0, & \text{otherwise, } \forall\ s \in s_l,\ a = 1, ..., M_l^{(2)}\ l = 1, ..., \sigma, \\ & k = 1, ..., B^{(2)}. \end{cases}$$

$$X_{im}^{(1)} = \begin{cases} 1, & \text{if sublot } i \text{ is formed at Stage 1 on machine } m \\ 0, & \text{otherwise, } \forall i = 1, ..., B^{(1)},\ m = 1, ..., M^{(1)}. \end{cases}$$

$$X_{lka}^{(2)} = \begin{cases} 1, & \text{if sublot } k \text{ of product type } l \text{ is formed on machine } a \text{ at Stage 2} \\ 0, & \text{otherwise, } \forall l = 1, ..., \sigma,\ k = 1, ..., B^{(2)},\ a = 1, ..., M_l^{(2)}. \end{cases}$$

6.1.1 Un-Capacitated Hybrid Flow Shop

As noted earlier, un-capacitated refers to an absence of a maximum or minimum restriction on batch sizes. Without these restrictions on the batch size, a batch at the first stage, can be of any size. In case the ready times of the jobs are the same, the first stage becomes relatively trivial; however, the second stage that includes sets of parallel machines remains quite complex. Since the first stage has un-capacitated batch size, Proposition 4.5.1 still holds for regular performance measures. Therefore, the column generation approach for all the un-capacitated problems in this section share the same method of passing the first stage completion time values for every soldier (job) to the second stage. The column generation approach for the capacitated problems will be different as we present later.

Decomposition II Formulation for Un-Capacitated Multi-Machine at Stage 1 of Hybrid Flow Shop

Minimize:

$$\sum_{l=1}^{\sigma}\sum_{s\in s_l}\left(\sum_{t=1}^{T}(-P^{(1)}-R_s)x_t^{(1)}+(C_{st}^{(2)}-P_l^{(2)})x_{tl}^{(2)}\right) \tag{6.1}$$

Subject to:

$$(\gamma^{(1)}) \quad \sum_{t=1}^{T} x_t^{(1)} = 1. \tag{6.2}$$

$$(\gamma_l^{(2)}) \quad \sum_{t=1}^{T} x_{tl}^{(2)} = 1, \quad \forall\, l = 1, ..., \sigma. \tag{6.3}$$

$$(\mu_s^{(1)}) \quad \sum_{t=1}^{T} \left((C_{st}^{(2)} - P_l^{(2)}) x_{tl}^{(2)} - C_{st}^{(1)} x_t^{(1)} \right) = w_s^{(2)}, \quad \forall\, s \in s_l,\ l = 1, ..., \sigma. \tag{6.4}$$

As discussed in Chapter 4, calculating the dual to the reformulation generates the objectives to the subproblems. The objectives of the two subproblems are very similar to the first and second stages of the improved reformulation of the AFGP in Chapter 4. The first stage objective determines the optimal schedule for jobs; however, there are multiple parallel machines available. Similar to the reformulation of the AFGP, there is an additional completion value added to the objective to enable the subproblem's calculation of the schedule. A schedule in this first stage has changed from enumerating over the number of possible sublots that reduces cost to enumerating over the possible number of machines. For each enumeration, the number of total sublots allowed is bounded with respect to the total number of jobs. Unlike the AFGP, there are no restrictions on the length of time a component can wait before being processed in the first stage.

Dual

Maximize:

$$\gamma^{(1)} + \sum_{l=1}^{\sigma} \gamma_l^{(2)} \tag{6.5}$$

Subject to:

$$(x^{(1)}) \quad -\mu_s^{(1)} C_{st}^{(1)} + \gamma^{(1)} \leq \sum_{l=1}^{\sigma} \sum_{s \in s_l} (-R_s - P^{(1)}), \tag{6.6}$$

$$(x^{(2)}) \quad \sum_{l=1}^{\sigma} \sum_{s \in s_l} \mu_s^{(1)} C_{stk}^{(2)} + \sum_{l=1}^{\sigma} \gamma_l^{(2)} \leq \sum_{l=1}^{\sigma} \sum_{s \in s_l} (C_{st}^{(2)} - P_l^{(2)}), \tag{6.7}$$

$$\mu_s^{(1)},\ \gamma^{(1)},\ \gamma_l^{(2)}\text{unrestricted.} \tag{6.8}$$

Pricing Problems

ZPP1:

Minimize

$$\sum_{l=1}^{\sigma}\sum_{s\in s_l}\left(\sum_{m=1}^{M^{(1)}}\sum_{i=1}^{B^{(1)}} C_{smi}^{(1)} - (R_s + P^{(1)}) + \mu_s^{(1)}\sum_{m=1}^{M^{(1)}}\sum_{i=1}^{B^{(1)}} C_{smi}^{(1)}\right) - \gamma^{(1)} \tag{6.9}$$

Subject to:

$$\begin{gathered}\sum_{m=1}^{M^{(1)}}\sum_{i=1}^{B^{(1)}} y_{smi}^{(1)} = 1,\\ \forall\ s\in s_l,\ l=1,...,\sigma.\end{gathered} \tag{6.10}$$

$$\begin{gathered}C_{smi}^{(1)} \le (P^{(1)} + R_s)y_{smi}^{(1)},\\ \forall\ s\in s_l,\ l=1,...,\sigma,\ m=1,...,M^{(1)},\ i=1,...,B^{(1)}.\end{gathered} \tag{6.11}$$

$$\begin{gathered}C_{smi}^{(1)} - (P^{(1)} + R_s)y_{smi}^{(1)} \ge 0,\\ \forall\ a=1,...,M^{(1)},\ i=1,...,B^{(1)},\ s\in s_l,\ l=1,...,\sigma.\end{gathered} \tag{6.12}$$

$$\begin{gathered}C_{s'mi}^{(1)} - C_{smi-1}^{(1)} - P^{(1)} + G(1-y_{s'mi}^{(1)}) \ge 0,\\ \forall\ m=1,...,M^{(1)},\ i=2,...,B^{(1)},\\ s'\in s_l,\ s\in s_l,\ l=1,...,\sigma,\ s'\ne s.\end{gathered} \tag{6.13}$$

$$\begin{gathered}C_{s'mi}^{(1)} - C_{smi}^{(1)} + G(1-y_{s'mi}^{(1)}) + G(1-y_{smi}^{(1)}) \ge 0,\\ \forall\ m=1,...,M^{(1)},\ i=1,...,B^{(1)},\\ s'\in s_l,\ s\in s_l,\ l=1,...,\sigma\ s'\ne s.\end{gathered} \tag{6.14}$$

The second stage objective is similar to the AFGP reformulation, only without some

of the dual terms reflecting the absence of the third stage. The schedule for this stage is still defined over the number of machines. At each iteration, the subproblem enumerates over the different possible machines, and selects the numbers of machines that reduce cost as well as their respective completion times.

ZPP2:

Minimize

$$\sum_{l=1}^{\sigma}\left(\sum_{s\in s_l}\sum_{a=1}^{M_l^{(2)}}\sum_{k=1}^{B^{(2)}}C_{sak}^{(2)}-(P_l^{(2)})-\sum_{a=1}^{M_l^{(2)}}\sum_{k=1}^{B^{(2)}}\mu_s^{(1)}(C_{sak}^{(2)}-P_l^{(2)})-\gamma_l^{(2)}\right) \tag{6.15}$$

Subject to:

Batch Job Assignment:

$$\begin{gathered}\sum_{a=1}^{M_l^{(2)}}\sum_{k=1}^{B^{(2)}}y_{sak}^{(2)}=1,\\ \forall\ s\in s_l,\ l=1,...,\sigma.\end{gathered} \tag{6.16}$$

Batch Minimum Size:

$$\begin{gathered}\sum_{s\in s_l}y_{sak}^{(2)}\geq z_{ak}L_l^{(2)},\\ \forall\ a=1,...,M_l^{(2)},\ k=1,...,B^{(2)},\ l=1,...,\sigma.\end{gathered} \tag{6.17}$$

Batch Maximum Size:

$$\begin{gathered}\sum_{s\in s_l}y_{sak}^{(2)}\leq z_{ak}U_l^{(2)},\\ \forall\ a=1,...,M_l^{(2)},\ k=1,...,B^{(2)},\ l=1,...,\sigma.\end{gathered} \tag{6.18}$$

Batch Order Precedence:

$$\begin{gathered}z_{ak}\leq z_{a(k-1)},\\ \forall\ a=1,...,M_l^{(2)},\ k=1,...,B^{(2)},\ l=1,...,\sigma.\end{gathered} \tag{6.19}$$

Batch Schedule Feasibility:

$$C^{(2)}_{sak} - (P^{(2)}_l + C^{(1)}_{si}) + G(1 - y^{(1)}_{smi}) + G(1 - y^{(2)}_{sak}) \geq 0, \\ \forall\ s \in s_l,\ a = 1, ..., M^{(2)}_l,\ k = 1, ..., B^{(2)},\ i = 1, ..., B^{(1)},\ m = 1, ..., M^{(1)},\ l = 1, ..., \sigma. \tag{6.20}$$

Batch Precedence:

$$C^{(2)}_{s'ak} - C^{(2)}_{sak-1} - P^{(2)}_l + G(1 - y^{(2)}_{s'ak}) \geq 0, \\ \forall\ a = 1, ..., M^{(2)}_l,\ k = 1, ..., B^{(2)},\ s' \in s_l,\ s \in s_l\ s' \neq s, \\ l = 1, ..., \sigma. \tag{6.21}$$

Batch Times Equality:

$$C^{(2)}_{s'ak} - C^{(2)}_{sak} + G(1 - y^{(2)}_{s'ak}) + G(1 - y^{(2)}_{sak}) \geq 0, \\ \forall\ a = 1, ..., M^{(2)}_l,\ k = 1, ..., B^{(2)},\ s' \in s_l,\ s \in s_l\ s' \neq s, \\ l = 1, ..., \sigma. \tag{6.22}$$

Completion Times:

$$C^{(2)}_{sak} \leq G(y^{(2)}_{sak}), \\ \forall\ s \in s_l,\ a = 1, ..., M^{(2)}_l,\ k = 1, ..., B^{(2)}, \\ l = 1, ..., \sigma. \tag{6.23}$$

MIP Formulation for Un-Capacitated Multi-Machine at Stage 1 for Hybrid Flow Shop

Minimize:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} (W^{(1)}_s + P^{(1)}) + \sum_{l=1}^{\sigma} \sum_{s \in s_l} (W^{(2)}_s + P^{(2)}_l) + \sum_{l=1}^{\sigma} \sum_{a=1}^{I^{(2)}} \sum_{k=1}^{B^{(2)}} \gamma_{la} X^{(2)}_{lka}$$

Stage 1

Sublot Assignment:

$$\sum_{m=1}^{M^{(1)}} \sum_{i=1}^{B^{(1)}} Z_{slim}^{(1)} = 1, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.24}$$

Sublot Formation Indicator Variable:

$$\sum_{m=1}^{M^{(1)}} X_{im}^{(1)} \leq 1, \quad \forall\ i = 1, ..., B^{(1)}. \tag{6.25}$$

Sublot Indicator Bound:

$$G(X_{im}^{(1)}) \geq S_{im}^{(1)}, \quad \forall\ i = 1, ..., B^{(1)}\ m = 1, ..., M^{(1)}. \tag{6.26}$$

Wait Time:

$$0 \leq C_{im}^{(1)} - P^{(1)} - R - G(1 - Z_{slim}^{(1)}) \leq W_s^{(1)}, \quad \forall\ s = 1, ..., ES,\ l = 1, ..., \sigma,\ i = 1, ..., B^{(1)},\ m = 1, ..., M^{(1)}. \tag{6.27}$$

Sublot Order:

$$C_{11}^{(1)} - P^{(1)} \geq 0, \tag{6.28}$$

$$C_{(i+1)m}^{(1)} - C_{im}^{(1)} - P^{(1)} \geq 0, \quad \forall\ i = 1, ..., B^{(1)},\ m = 1, ..., M^{(1)\cdot} \tag{6.29}$$

Stage 1 Completion Times:

$$C_{11}^{(1)} - P^{(1)} - R_s + G(1 - Z_{sl11}^{(1)}) \geq 0, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma; \tag{6.30}$$

$$C_{im}^{(1)} - P^{(1)} - R_s + G(1 - Z_{slim}^{(1)}) \geq 0,$$
$$\forall\ i = 2, ..., B^{(1)} - 1,\ m = 1, ..., M^{(1)},\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.31}$$

Stage 2

$$\sum_{k=1}^{B_l^{(2)}} \sum_{a=1}^{M_l^{(2)}} Z_{slka}^{(2)} = 1,$$
$$\forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.32}$$

Sublot Size:

$$\sum_{s \in s_l} Z_{slka}^{(2)} = S_{lka}^{(2)},$$
$$\forall\ l = 1, ..., \sigma,\ k = 1, ..., B_l^{(2)}, a = 1, ..., M_l^{(2)}. \tag{6.33}$$

Sublot formation indicator value:

$$\sum_{a=1}^{M_l^{(2)}} X_{lka}^{(2)} \leq 1,$$
$$\forall\ l = 1, ..., \sigma,\ k = 1, ..., B_l^{(2)}. \tag{6.34}$$

Sublot Precedence:

$$\sum_{a=1}^{M_l^{(2)}} X_{lka}^{(2)} \leq \sum_{a=1}^{M_l^{(2)}} X_{l(k-1)a}^{(2)},$$
$$\forall\ l = 1, ..., \sigma,\ k = 2, ..., B_l^{(2)}. \tag{6.35}$$

Minimum Sublot Size:

$$L^{(2)}(X_{lka}^{(2)}) \leq S_{lka}^{(2)},$$
$$\forall\ l = 1, ..., \sigma,\ k = 1, ..., B_l^{(2)}\ a = 1, ..., M_l^{(2)}. \tag{6.36}$$

Maximum Sublot Size:

$$\begin{aligned} U^{(2)}(X^{(2)}_{lka}) \geq S^{(2)}_{lka}, \\ \forall\ l = 1, ..., \sigma,\ k = 1, ..., B^{(2)}_l\ a = 1, ..., M^{(2)}_l. \end{aligned} \tag{6.37}$$

Stage 2 Wait Time:

$$\begin{aligned} 0 \leq C^{(2)}_{lka} - P^{(2)}_l - C^{(1)}_i - G(1 - Y^{(2)}_{slika}) \leq W^{(2)}_s, \\ \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., B^{(1)}, \\ k = 1, ..., B^{(2)}_l,\ a = 1, ..., M^{(2)}_l. \end{aligned} \tag{6.38}$$

The following constraints ensure that a sublot cannot use the same machine until the previous sublot is finished. The first constraint serves as initialization.

$$\begin{aligned} C^{(2)}_{l11} - C^{(2)}_{l01} - P^{(2)}_l + G(1 - X^{(2)}_{l11}) \geq 0, \\ \forall\ l = 1, ..., \sigma. \end{aligned} \tag{6.39}$$

$$\begin{aligned} C^{(2)}_{lka} - C^{(2)}_{lk'a} - P^{(2)}_l + G(1 - X^{(2)}_{lka}) \geq 0, \\ \forall\ l = 1, ..., \sigma,\ k = 2, ..., B^{(2)}_l,\ k' = 1, ..., B^{(2)}_l,\ k' < k \\ a = 1, ..., M^{(2)}_l. \end{aligned} \tag{6.40}$$

Sublot order Precedence:

$$\begin{aligned} C^{(2)}_{lka} - C^{(2)}_{lk'a'} + G(1 - X^{(2)}_{lka}) \geq 0, \\ \forall\ l = 1, ..., \sigma,\ k = 2, ..., B^{(2)}_l,\ k' = 1, ..., B^{(2)}_l,\ k' < k, \\ a = 1, ..., M^{(2)}_l, a' = 1, ...M^{(2)}_l. \end{aligned} \tag{6.41}$$

The following constraint assures the batches of size zero have zero completion times:

$$\begin{aligned} C^{(2)}_{lka} \leq G(X^{(2)}_{lka}), \\ \forall\ k = 1, ..., B^{(2)}_l,\ a = 1, ..., M^{(2)}_l,\ l = 1, ..., \sigma. \end{aligned} \tag{6.42}$$

Symmetry Breaking Constraints:

$$\sum_{k=1}^{B_l^{(2)}} X_{lk(a-1)}^{(2)} \geq \sum_{k=1}^{B_l^{(2)}} X_{lka}^{(2)}, \quad \forall\ a = 2, ..., M_l^{(2)},\ l = 1, ..., \sigma. \tag{6.43}$$

Note that the binary variable $Y_{slika}^{(2)}$, which is a multiplication of variables $Z_{slka}^{(2)}$ and $Z_{sli}^{(1)}$, can be linearized by using the following inequalities:

$$Z_{slka}^{(2)} \geq \sum_{i=1}^{B^{(1)}} Y_{slika}^{(2)}, \quad \forall\ s \in s_l, \quad a = 1, ..., M_l^{(2)},\ k = 1, ..., B_l^{(2)},\ l = 1, ..., \sigma. \tag{6.44}$$

$$Z_{sli}^{(1)} \geq \sum_{k=1}^{B_l^{(2)}} \sum_{a=1}^{I_l^{(2)}} Y_{slika}^{(2)}, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., B^{(1)}. \tag{6.45}$$

$$Z_{sli}^{(1)} + Z_{slka}^{(2)} - 1 \leq Y_{slika}^{(2)}, \quad \forall\ s \in s_l,\ i = 1, ..., B^{(1)}, \quad a = 1, ..., M_l^{(2)},\ k = 1, ..., B_l^{(2)},\ l = 1, ..., \sigma, . \tag{6.46}$$

Stage 2: Batch Completion Times

$$C_{l1a}^{(2)} - P_l^{(2)} - C_i^{(1)} + G(1 - Y_{sli1a}^{(2)}) \geq 0, \quad \forall\ s \in s_l,\ i = 1, ..., B^{(1)},\ a = 1, ..., M_l^{(2)},\ l = 1, .., \sigma \tag{6.47}$$

$$C_{lka}^{(2)} - P_l^{(2)} - C_i^{(1)} + G(1 - Y_{slika}^{(2)}) \geq 0, \quad \forall\ s \in s_l,\ i = 1, ..., B^{(1)},\ a = 1, ..., M_l^{(2)},\ k = 1, ..., B_l^{(2)},\ l = 1, ..., \sigma \tag{6.48}$$

Equal Ready Times

In this version of the hybrid flow shop model, we address the two-stage parallel machine problem for equal ready times for all the jobs in the system. As an un-capacitated first stage, the parallel machines in the first stage have no restriction on the size of the batches formed. We do not specify that each job has to arrive at time zero but simply they all arrive at the same time. With equal ready times, the solution method is slightly different than the Sequential approach that we use for most of the the un-capacitated problems that we present. In our investigation for a solution method for the AFGP problem, the unequal ready times were the main issue behind the problem not being able to decompose into three subproblems without having to pass information from the first stage to the second. In the case where ready times are equal, the Decomposition II method solves the problem optimally.

Results

We present the performance results of the Decomposition II approach and direct solution by CPLEX for the equal ready time case for the un-capacitated, multi-machine first stage, hybrid flow shop in Figure 6.2. We ran ten scenarios for each model variation. The ten scenarios represent ten different sets of processing times for Stage 2. The model variations changed either by number of job types or the number of jobs. The performance of the Decomposition II method and direct solution solved using CPLEX have relatively similar performances except for the large-case, small job-type model variation (2T2M15J). When the number of jobs increases for the two job-type case, the average CPU times required by CPLEX to solve the solution directly increases dramatically; however, for the same number of jobs for the scenario with four job-types, the average CPU times required by CPLEX to solve directly reduces significantly. This resulting spike behavior is prevalent first in Figure 5.3 and will be seen in almost all the charts in this chapter. The behavior we see is graphically displaying the observation we made in Chapter 3, where when the number of soldiers were

fixed, but the number of skill sets increased, the required CPU time decreased. For these similar scenarios we see this relationship in our results for jobs and job types. For a fixed number of jobs, the CPU time decreases with the increase of job types. As the number of jobs increases, the relationship becomes clearer as the spike in Figure 6.2 indicates.

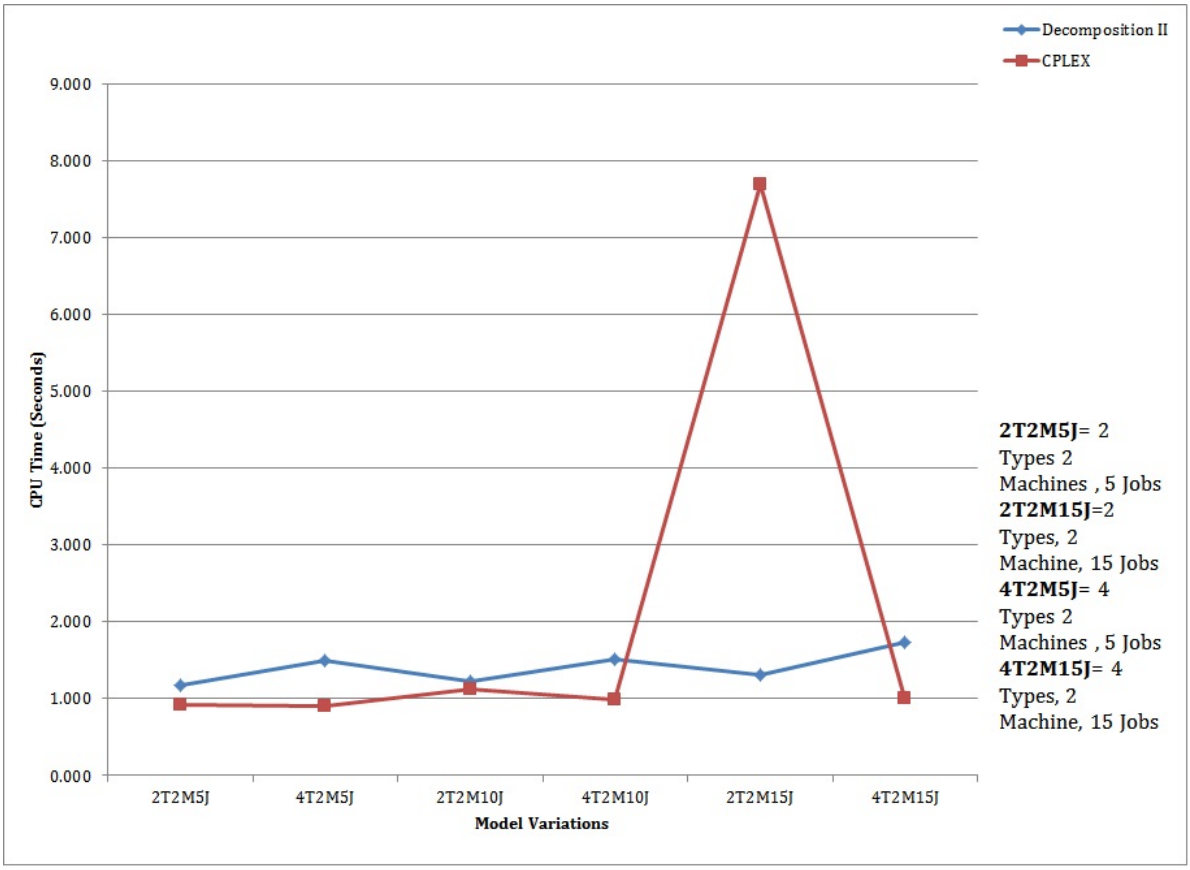


Figure 6.2: Un-Capacitated Multi-Machine First Stage, Equal Ready Times, Hybrid Flow Shop: Average performance values for solutions obtained by Decomposition II and directly using CPLEX

Unequal Ready Times

This particular model for unequal ready times of the first-stage un-capacitated models shown in Figure 6.1 is the most complex. With unequal ready times and unequal sublots or more specifically variable sublots, the jobs can form sublots of any size at Stage 2 under individual ready times for forming the sublots at the first stage. Similar to the previous model with

unequal sublot sizes and equal ready times, the first stage is setup with m-parallel machines that process various component types, which are allowed to mix to form a sublot before being separated by their types at the second stage. Since there are only two stages, the column generation of the formulation is relatively simple, given the complexity of a multi-parallel machine problem. With two subproblems, there are binary decision variables for each subproblem, which select the best schedule at each iteration for the two stages. The distinct decision variables $x^{(1)}$ and $x^{(2)}$ permit different iterations over schedules to be chosen. Therefore, the master problem must enforce that only feasible combinations of schedules from Stage 1 and Stage 2 are selected. This feasibility check is obtained with the constraint that enforces the second stage completion time chosen to be larger than the first stage completion time plus the second stage processing time.

Table 6.1: Hybrid Flow Shop Parameters

Parameter	Value
$B^{(1)}$	3
$P^{(1)}$	2
$P_l^{(2)}$	3,2,2,2
$B^{(2)}$	3
$M^{(2)}$	2
$L^{(2)}$	1
$U^{(2)}$	5

The parameters for the hybrid flow shop are presented in Table 6.1.

Results

In this section, we present a comparison between the Decomposition II approach and solutions obtained directly using CPLEX for the un-capacitated multi-machine Stage 1 for the hybrid flow shop problem as described previously. For both solution methods, we ran ten scenarios for each model variation. In each model variation, we changed either the number of job types or the number of jobs. The performance results are shown in Figure 6.3. The

Decomposition II method has relatively consistent performance for both average CPU time required and accuracy compared with that of the optimum seeking method. For every model variation, the Decomposition II method required on average less than 5 seconds. The accuracy of the solution method is shown by the Decomposition II Gap line that only ranges a total of 0.17% from the biggest to smallest gap percentage. The percentage gap between the Decomposition II method solution and the optimal solution is consistently around 2%. The solutions obtained directly by CPLEX require on average about the same CPU time as the Decomposition II method for smaller models that include only five jobs; however, for the two job type case with 15 jobs, the direct solution required on average almost 33 seconds whereas Decomposition II only required on average 1.5 seconds. As the job type increased for the same number of jobs, the average CPU time required by solving directly with CPLEX reduced to about 12 seconds but was still greater than the average time required by the Decomposition II method. This behavior that results in a spike of CPU time for CPLEX is consistent with what we observed for the AFGP in Figure 5.3, that is as the number of job types increase for the same number of jobs, the average CPU time reduces as the problem naturally decomposes in Stage 2. With similarly structured scenarios, this behavior will be evident in all of our extension results in the following sections. The performance of Decomposition II is also consistent with the results reported in Chapter 5 for the AFGP test problems, where for small-sized problem instances it performs about the same as the solution obtained directly by CPLEX, but whose average CPU time requirements grow at a much slower rate than the solution obtained directly.

6.1.2 Capacitated Hybrid Flow Shop

In this section, we present two formulations of the two-stage hybrid flow shop with a capacitated single machine at Stage 1. This model configuration is similar to the AFGP Stages 1 and 2 but with a restriction on the size of sublot that can be formed at the single machine

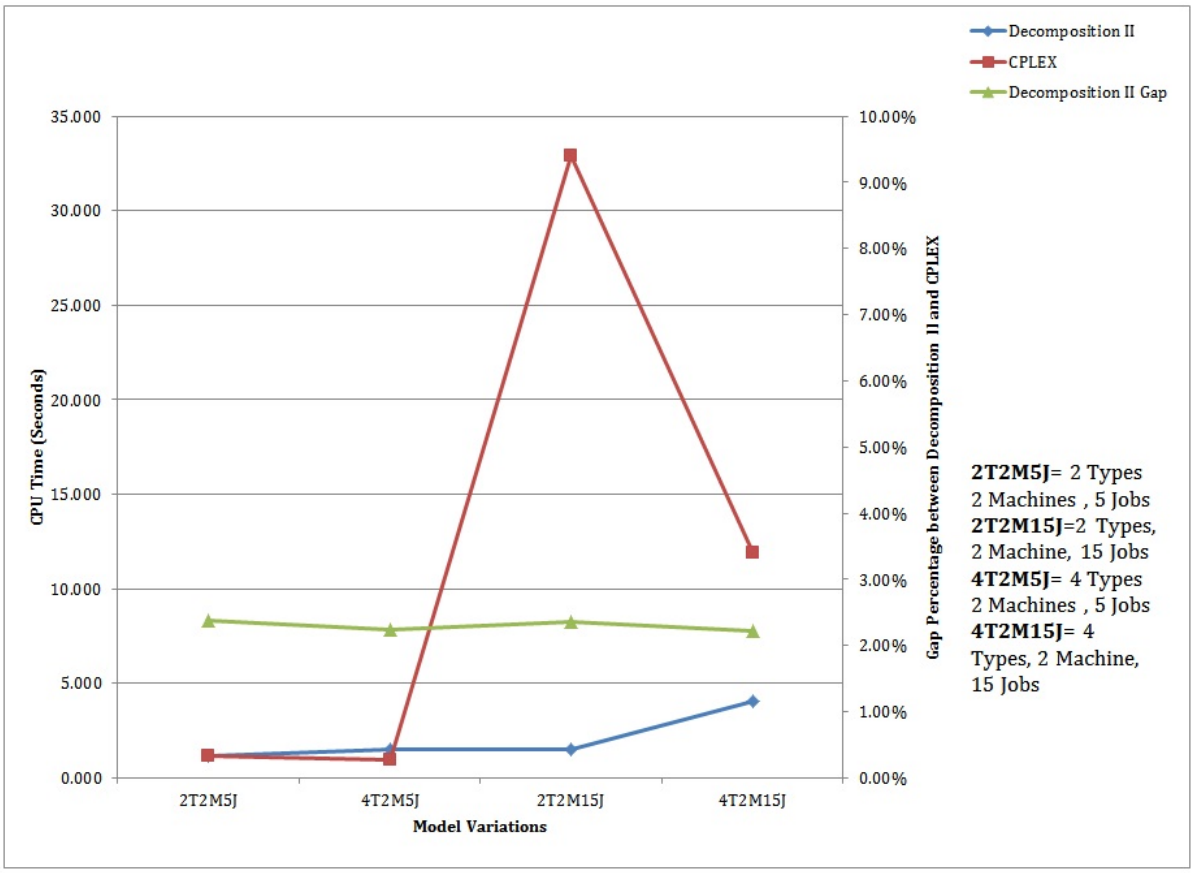


Figure 6.3: Un-Capacitated Multi-Machine First Stage, Hybrid Flow Shop: Average performance values for solutions obtained by Decomposition II and directly using CPLEX

at Stage 1. We solve the problem for equal ready times and unequal ready times using the Decomposition II method presented in Chapter 4 and by solving the problem directly using CPLEX. Formulations for both the Decomposition II and MIP formulations are presented next.

Decomposition II Formulation

Minimize:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} \left(\sum_{t=1}^{T} (-P^{(1)} - R_s) x_t^{(1)} + (C_{st}^{(2)} - P_l^{(2)}) x_{tl}^{(2)} \right) \tag{6.49}$$

Subject to:

$$(\gamma^{(1)}) \quad \sum_{t=1}^{T} x_t^{(1)} = 1. \tag{6.50}$$

$$(\gamma_l^{(2)}) \quad \sum_{t=1}^{T} x_{tl}^{(2)} = 1, \quad \forall\, l = 1, ..., \sigma. \tag{6.51}$$

Complicating Constraints

$$(\mu_s^{(1)}) \quad \sum_{t=1}^{T} \left((C_{st}^{(2)} - P_l^{(2)}) x_{tl}^{(2)} - C_{st}^{(1)} x_t^{(1)} \right) = w_s^{(2)}, \quad \forall\, l = 1, ..., \sigma,\ s \in s_l. \tag{6.52}$$

Dual

Maximize:

$$\gamma^{(1)} + \sum_{l=1}^{\sigma} \gamma_l^{(2)}. \tag{6.53}$$

Subject to:

$$(x^{(1)}) \quad -\mu_s^{(1)} C_{st}^{(1)} + \gamma^{(1)} \leq \sum_{s=1}^{S} -R_s - P^{(1)}. \tag{6.54}$$

$$(x^{(2)}) \quad \sum_{l=1}^{\sigma} \sum_{s \in s_l} \mu_s^{(1)} C_{st}^{(2)} + \sum_{l=1}^{\sigma} \gamma_l^{(2)} \leq (C_{st}^{(2)} - P_l^{(2)}). \tag{6.55}$$

$$\mu_s^{(1)},\ \gamma^{(1)},\ \gamma_l^{(2)} \quad \text{unrestricted.} \tag{6.56}$$

Pricing Problems:

ZPP1:

Minimize:

$$\sum_{s=1}^{S}\left(\sum_{i=1}^{j}C_{si}^{(1)}-(R_s+P^{(1)})+\mu_s^{(1)}\sum_{i=1}^{j}C_{si}^{(1)}\right)-\gamma^{(1)} \tag{6.57}$$

Subject to:

$$\sum_{i=1}^{j}y_{si}^{(1)}=1,\quad \forall\ s=1,...,S. \tag{6.58}$$

$$\sum_{s=1}^{S}y_{si}^{(1)}\geq L^{(1)},\quad \forall\ i=1,...,j. \tag{6.59}$$

$$\sum_{s=1}^{S}y_{si}^{(1)}\leq U^{(1)},\quad \forall\ i=1,...,j. \tag{6.60}$$

$$C_{si}^{(1)}\leq (P^{(1)}+R_s)y_{si}^{(1)},\quad \forall\ s=1,...,S,\ i=1,...,j. \tag{6.61}$$

$$C_{si}^{(1)}-(P^{(1)}+R_s)y_{si}^{(1)}\geq 0,\quad \forall\ i=1,...,j,\ s=1,...,S. \tag{6.62}$$

$$C_{s'i}^{(1)}-C_{si-1}^{(1)}-P^{(1)}+G(1-y_{s'i}^{(1)})\geq 0,\quad \forall\ i=2,...,j,\ s'=1,...,S,\ s=1,...,S,\ s'\neq s. \tag{6.63}$$

$$C_{s'i}^{(1)} - C_{si}^{(1)} + G(1 - y_{s'i}^{(1)}) + G(1 - y_{si}^{(1)}) \geq 0, \\ \forall\ i = 1, ..., j,\ s' = 1, ..., S,\ s = 1, ..., S,\ s' \neq s. \tag{6.64}$$

ZPP2:

Minimize:

$$\sum_{l=1}^{\sigma} \left(\sum_{s \in s_l} \sum_{a=1}^{A_l} \sum_{k=1}^{B^{(2)}} C_{sak}^{(2)} - (P_l^{(2)}) - \sum_{a=1}^{A_l} \sum_{k=1}^{B^{(2)}} \mu_s^{(1)} (C_{sak}^{(2)} - P_l^{(2)}) - \gamma_l^{(2)} \right) \tag{6.65}$$

Subject to:

Batch Assignment:

$$\sum_{a=1}^{M_l} \sum_{k=1}^{B^{(2)}} y_{sak}^{(2)} = 1, \\ \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.66}$$

Batch Minimum Size:

$$\sum_{s \in s_l} y_{sak}^{(2)} \geq z_{ak} L_l^{(2)}, \\ \forall\ a = 1, ..., M_l,\ k = 1, ..., B^{(2)},\ l = 1, ..., \sigma. \tag{6.67}$$

Batch Maximum Size:

$$\sum_{s \in s_l} y_{sak}^{(2)} \leq z_{ak} U_l^{(2)}, \\ \forall\ a = 1, ..., M_l,\ k = 1, ..., B^{(2)},\ l = 1, ..., \sigma. \tag{6.68}$$

Batch Order Precedence:

$$z_{ak} \leq z_{a(k-1)}, \\ \forall\ a = 1, ..., M_l,\ k = 1, ..., B^{(2)},\ l = 1, ..., \sigma. \tag{6.69}$$

Batch Schedule Feasibility:

$$C_{sak}^{(2)} - (P_l^{(2)} + P^{(1)} + R_s) + G(1 - y_{sak}^{(2)}) \geq 0, \\ \forall\ s \in s_l,\ l = 1, ..., \sigma,\ a = 1, ..., M_l,\ k = 1, ..., B^{(2)}. \tag{6.70}$$

Batch Precedence:

$$C_{s'ak}^{(2)} - C_{sak-1}^{(2)} - P_l^{(2)} + G(1 - y_{s'ak}^{(2)}) \geq 0, \\ \forall\ a = 1, ..., M_l,\ k = 1, ..., B^{(2)},\ s' \in s_l,\ s \in s_l\ s' \neq s, \\ l = 1, ..., \sigma. \tag{6.71}$$

Batch Times Equality:

$$C_{s'ak}^{(2)} - C_{sak}^{(2)} + G(1 - y_{s'ak}^{(2)}) + G(1 - y_{sak}^{(2)}) \geq 0, \\ \forall\ a = 1, ..., M_l,\ k = 1, ..., B^{(2)},\ s' \in s_l,\ s \in s_l\ s' \neq s, \\ l = 1, ..., \sigma. \tag{6.72}$$

Completion Times:

$$C_{sak}^{(2)} \leq G(y_{sak}^{(2)}), \\ \forall\ s \in s_l,\ a = 1, ..., M_l,\ k = 1, ..., B^{(2)}, \\ l = 1, ..., \sigma. \tag{6.73}$$

MIP Formulation

Minimize:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} (W_s^{(1)} + P^{(1)}) + \sum_{l=1}^{\sigma} \sum_{s \in s_l} (W_s^{(2)} + P_l^{(2)}) + \sum_{l=1}^{\sigma} \sum_{a=1}^{M^{(2)}} \sum_{k=1}^{B^{(2)}} \gamma_{la} X_{lka}^{(2)}$$

Stage 1

Sublot Assignment:

$$\sum_{i=1}^{B^{(1)}} Z_{sli}^{(1)} = 1, \\ \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.74}$$

Minimum Sublot Size for Stage 1:

$$\begin{aligned} L^{(1)}(X_i^{(1)}) \leq S_i^{(1)}, \\ \forall\ i = 1, ..., B^{(1)}. \end{aligned} \tag{6.75}$$

Maximum Sublot Size for Stage 1:

$$\begin{aligned} U^{(1)}(X_i^{(1)}) \leq S_i^{(1)}, \\ \forall\ i = 1, ..., B^{(1)}. \end{aligned} \tag{6.76}$$

Wait Time:

$$\begin{aligned} 0 \leq C_i^{(1)} - P^{(1)} - R - G(1 - Z_{sli}^{(1)}) \leq W_s^{(1)}, \\ \forall\ s = 1, ..., S,\ l = 1, ..., \sigma,\ i = 1, ..., B^{(1)}. \end{aligned} \tag{6.77}$$

Sublot Order:

$$C_1^{(1)} - P^{(1)} \geq 0, \tag{6.78}$$

$$\begin{aligned} C_{i+1}^{(1)} - C_i^{(1)} - P^{(1)} \geq 0, \\ \forall\ i = 1, ..., B^{(1)}; \end{aligned} \tag{6.79}$$

Completion Times:

$$\begin{aligned} C_1^{(1)} - P^{(1)} - R_s + G(1 - Z_{sl1}^{(1)}) \geq 0, \\ \forall\ s \in s_l,\ l = 1, ..., \sigma; \end{aligned} \tag{6.80}$$

$$\begin{aligned} C_i^{(1)} - P^{(1)} - R_s + G(1 - Z_{sli}^{(1)}) \geq 0, \\ \forall\ i = 2, ..., B^{(1)} - 1,\ s \in s_l,\ l = 1, ..., \sigma. \end{aligned} \tag{6.81}$$

Stage 2

$$\sum_{k=1}^{B_l^{(2)}} \sum_{a=1}^{I_l^{(2)}} Z_{slka}^{(2)} = 1, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.82}$$

Sublot Size:

$$\sum_{s \in s_l} Z_{slka}^{(2)} = S_{lka}^{(2)}, \quad \forall\ l = 1, ..., \sigma,\ k = 1, ..., B_l^{(2)}, a = 1, ..., M_l^{(2)}. \tag{6.83}$$

Sublot formation indicator value:

$$\sum_{a=1}^{I_l^{(2)}} X_{lka}^{(2)} \leq 1, \quad \forall\ l = 1, ..., \sigma,\ k = 1, ..., B_l^{(2)}. \tag{6.84}$$

Sublot Precedence:

$$\sum_{a=1}^{I_l^{(2)}} X_{lka}^{(2)} \leq \sum_{a=1}^{M_l^{(2)}} X_{l(k-1)a}^{(2)}, \quad \forall\ l = 1, ..., \sigma,\ k = 2, ..., B_l^{(2)}. \tag{6.85}$$

Minimum Sublot Size:

$$L^{(2)}(X_{lka}^{(2)}) \leq S_{lka}^{(2)}, \quad \forall\ l = 1, ..., \sigma,\ k = 1, ..., B_l^{(2)}\ a = 1, ..., M_l^{(2)}. \tag{6.86}$$

Maximum Sublot Size:

$$\begin{aligned} U^{(2)}(X^{(2)}_{lka}) \geq S^{(2)}_{lka}, \\ \forall\ l = 1, ..., \sigma,\ k = 1, ..., B^{(2)}_l\ a = 1, ..., M^{(2)}_l. \end{aligned} \tag{6.87}$$

Wait Time:

$$\begin{aligned} 0 \leq C^{(2)}_{lka} - P^{(2)}_l - C^{(1)}_i - G(1 - Y^{(2)}_{slika}) \leq W^{(2)}_s, \\ \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., B^{(1)}, \\ k = 1, ..., B^{(2)}_l,\ a = 1, ..., M^{(2)}_l. \end{aligned} \tag{6.88}$$

The following constraints ensure that a sublot cannot use the same machine until the previous sublot is finished. The first constraint serves as initialization.

$$\begin{aligned} C^{(2)}_{l11} - C^{(2)}_{l01} - P^{(2)}_l + G(1 - X^{(2)}_{l11}) \geq 0, \\ \forall\ l = 1, ..., \sigma. \end{aligned} \tag{6.89}$$

$$\begin{aligned} C^{(2)}_{lka} - C^{(2)}_{lk'a} - P^{(2)}_l + G(1 - X^{(2)}_{lka}) \geq 0, \\ \forall\ l = 1, ..., \sigma,\ k = 2, ..., B^{(2)}_l,\ k' = 1, ..., B^{(2)}_l,\ k' < k, \\ a = 1, ..., M^{(2)}_l. \end{aligned} \tag{6.90}$$

Sublot order Precedence:

$$\begin{aligned} C^{(2)}_{lka} - C^{(2)}_{lk'a'} + G(1 - X^{(2)}_{lka}) \geq 0, \\ \forall\ l = 1, ..., \sigma,\ k = 2, ..., B^{(2)}_l,\ k' = 1, ..., B^{(2)}_l,\ k' < k, \\ a = 1, ..., I^{(2)}_l, a' = 1, ...M^{(2)}_l. \end{aligned} \tag{6.91}$$

The following constraint assures the batches of size zero have zero completion times:

$$\begin{aligned} C^{(2)}_{lka} \leq G(X^{(2)}_{lka}), \\ \forall\ k = 1, ..., B^{(2)}_l,\ a = 1, ..., M^{(2)}_l,\ l = 1, ..., \sigma. \end{aligned} \tag{6.92}$$

Symmetry Breaking Constraints:

$$\sum_{k=1}^{B_l^{(2)}} X_{lk(a-1)}^{(2)} \geq \sum_{k=1}^{B_l^{(2)}} X_{lka}^{(2)}, \quad \forall\ a = 2, ..., M_l^{(2)},\ l = 1, ..., \sigma. \tag{6.93}$$

Note that the binary variable $Y_{slika}^{(2)}$, which is a multiplication of variables $Z_{slka}^{(2)}$ and $Z_{sli}^{(1)}$, can be linearized by using the following inequalities:

$$Z_{slka}^{(2)} \geq \sum_{i=1}^{B^{(1)}} Y_{slika}^{(2)}, \quad \forall\ s \in s_l, \quad a = 1, ..., I_l^{(2)},\ k = 1, ..., B_l^{(2)},\ l = 1, ..., \sigma. \tag{6.94}$$

$$Z_{sli}^{(1)} \geq \sum_{k=1}^{B_l^{(2)}} \sum_{a=1}^{M_l^{(2)}} Y_{slika}^{(2)}, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., B^{(1)}. \tag{6.95}$$

$$Z_{sli}^{(1)} + Z_{slka}^{(2)} - 1 \leq Y_{slika}^{(2)}, \quad \forall\ s \in s_l,\ i = 1, ..., B^{(1)}, \quad a = 1, ..., M_l^{(2)},\ k = 1, ..., B_l^{(2)},\ l = 1, ..., \sigma. \tag{6.96}$$

Batch Completion Times

$$C_{l1a}^{(2)} - P_l^{(2)} - C_i^{(1)} + G(1 - Y_{sli1a}^{(2)}) \geq 0, \quad \forall\ s \in s_l,\ i = 1, ..., B^{(1)},\ a = 1, ..., M_l^{(2)},\ l = 1, .., \sigma. \tag{6.97}$$

$$C_{lka}^{(2)} - P_l^{(2)} - C_i^{(1)} + G(1 - Y_{slika}^{(2)}) \geq 0, \quad \forall\ s \in s_l,\ i = 1, ..., B^{(1)},\ a = 1, ..., M_l^{(2)},\ k = 1, ..., B_l^{(2)},\ l = 1, ..., \sigma. \tag{6.98}$$

Equal Ready Times

For the equal ready time case, the model formulation is the same. We account for the equal ready time by setting the ready time parameter for every job to be the same as that for the Unequal Ready Time case. For testing purposes, we use a set of ten different processing time values for Stage 2 for performance comparisons with solutions obtained directly through CPLEX.

Results

The performance results for the Decomposition II method and solutions obtained directly by CPLEX for the capacitated single-machine, first stage hybrid flow shop are shown in Figure 6.4. For each model variation, ten scenarios were run using a set of ten different processing times. The performance of Decomposition II is fairly consistent for each of the model variations that vary for job types and the number of jobs. The performance of CPLEX, as it solves the solutions directly, for this model configuration does not have the same behavior we have observed previously. While there is still a significant increase for the $2T2M15J$ case, there is an even larger increase for the $4T2M15J$ case. For this equal ready time case, there are several alternate solutions, which enforces the model to search over several allocations of jobs to sublots in the first stage, thus affecting the completion times of the second stage.

Unequal Ready Times

The case of both stages being capacitated, having unequal ready times of the jobs, is computationally the hardest of all the problems shown in Figure 6.1. In this instance of the hybrid flow shop problem, the first stage a single machine with a restriction on the batch size the machine can handle. The individual jobs that intermingle in the first stage consists of can

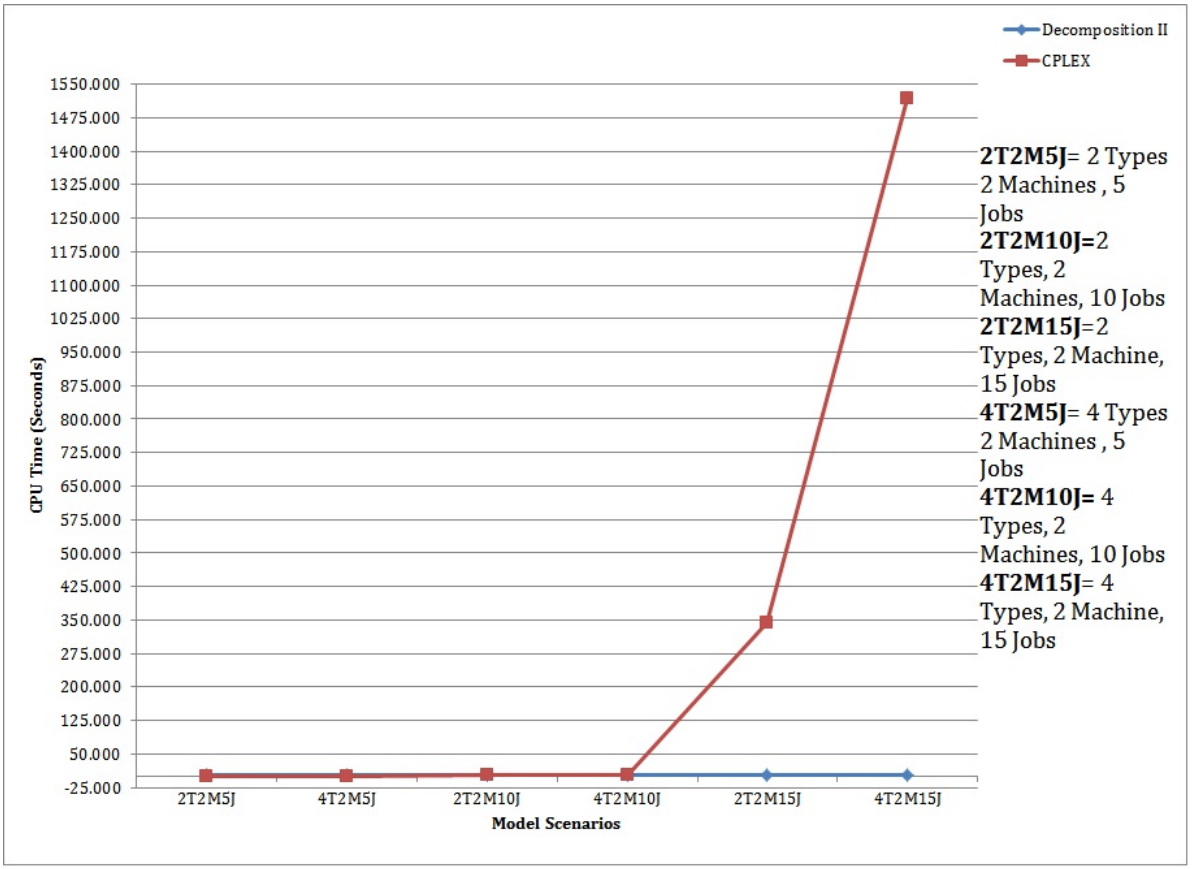


Figure 6.4: Capacitated Single-Machine First Stage, Equal Ready Times, Hybrid Flow Shop: Average performance values for solutions obtained by Decomposition II and directly using CPLEX

arrive at distinct times. The sizes of the batches created are different and they vary from stage to stage. We use the Decomposition II method to solve this problem as presented in Chapter 4 where information from Stage 1 is passed to Stage 2. The first stage generates schedules by enumerating over the number of classes to obtain the most reduced cost. The second stage will generate schedules in the subproblem by enumerating over the number of machines to obtain the most reduced cost. The second subproblem will be given a bound on the number of sublots that can be formed per machine to be consider while determining the schedule with the optimal number of machines per stage. A schedule for each job, and the completion times of jobs at each stage are determined in the subproblem. The model formulation presented for Decomposition II is used to solve this problem where the ready time parameters for individual jobs are unequal.

Table 6.2: Additional Hybrid Flow Shop Parameters

Parameter	Value
$L^{(1)}$	1
$U^{(1)}$	5

The parameters for the capacitated hybrid flow shop are the same as those presented in Table 6.1 with the additions listed in Table 6.2 to account for the restrictions on the sublot size in Stage 1.

Results

To compare the two formulations for the capacitated single machine at Stage 1 for the hybrid flow shop with unequal ready times, we performed several test runs for different model variations. Model variations include changes in the number of component types and the number of jobs. For each model variation, the model is run ten times for ten different sets of ready times. The average of the CPU times required and flow times for solutions obtained by Decomposition II and solutions obtained directly using CPLEX are presented in Figure 6.5. The average gap percentages between the solutions obtained by the Decomposition II method and those obtained directly by CPLEX are presented in Figure 6.5; however, since the gap percentage is 0% for each model variation, the gap percentage line hovers over the horizontal axis. Based on these performance results, the Decomposition II is very accurate compared to the optimal. The Decomposition II method, as compared with the CPLEX method requires roughly the same time for small-sized problem instance such as five jobs. For fewer job types, the Decomposition II method requires less average CPU time than that required for direct solution by CPLEX. The most CPU time required on average by CPLEX of all the model variations is the two product type, two machine, 15 job model (2T2M15J). The average amount of CPU time required for this model is 17.47 seconds for solutions obtained directly using CPLEX. In contrast, solutions obtained using the Decomposition II

method on average required CPU time of 1.78 seconds for the same model variation. As observed for the AFGP, the solutions obtained directly using CPLEX require less CPU time on average for the same number of jobs but with increased job types. For the last model variation, four job types, two machines and 15 jobs (4T2M15J), solutions obtained directly using CPLEX required on average less CPU time; however, the average CPU time required for solutions obtained directly is still greater than the average CPU time required by the Decomposition II method.

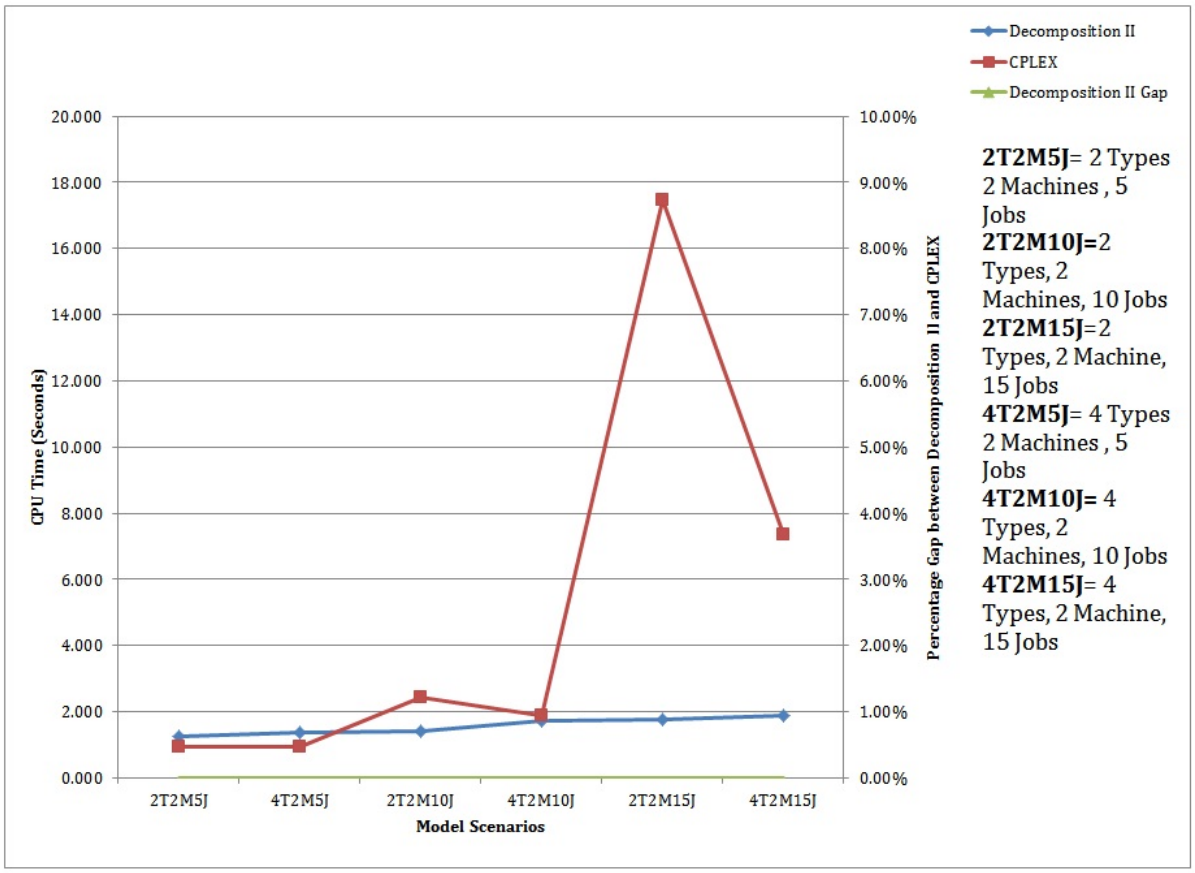


Figure 6.5: Capacitated Single Machine First Stage, Hybrid Flow Shop: Average performance values for solutions obtained by Decomposition II and directly using CPLEX

6.2 Agile Manufacturing

Agile manufacturing encompasses a wide scope of perspectives to improve the agility of companies. The common thread in the literature related to agile manufacturing is increasing competitiveness in an environment of continuous change [64]. Since there are several ways a firm can improve its ability to handle change, the literature reflects the myriad possibilities in the breadth of its content. Thus far, in the brief review of literature that follows, there have been few mathematical formulations used to improve a manufacturing system's agility.

Devor, Graves and Mills [30] present the outlook of agile manufacturing research in their paper. They give a history of the need of research for a kind of manufacturing system that allows companies to continue to thrive through continuous change. Agile manufacturing institutes have been setup at various universities through the coordination of academia, government and the U.S. Industry called Advanced Manufacturing Research Institute (AMRI). Each of the AMRI has a particular field of interest, and it develops research within its area but with applications across all the fields.

Quinn *et al.* [60] present a workcell design in an agile manufacturing as the ability to rapidly adapt to changing customer needs. As opposed to approaching the management and organizational shifts to adjust, the focus is on the dexterity of the machines. From this perspective, the design of a workcell is developed that is more flexible to change.

Lee [50] explores the agile manufacturing environment by focusing on reducing lead times for a variety of components and achieving dynamic reconfigurations of manufacturing systems. The significance of these two areas as part of the recommended strategies necessary for agile manufacturing has been reported in the literature. Lee investigates the design of manufacturing systems, and components, and their integration for agility in reducing lead times of product production. By first creating a design rule based on precedence constraints to determine the sequence of assemblies, a reduction in lead times for the manufacturing of

different products is reported. A mathematical formulation to reconfigure the manufacturing system with respect to changes of the product model is also developed. An integration of both approaches enables a manufacturing system to be more agile by being more sensitive to changing demand.

Gunasekaran [35] presents the first survey of agile manufacturing. A framework for an agile manufacturing system with all the relevant criteria is developed. The work reported in Agile Manufacturing is classified into four categories: strategies, technologies, systems, and people. The strategic category, includes the long-term operational policies that companies can use to achieve agile manufacturing. The next category, technologies, encompasses the hardware and software technology enabling agile manufacturing. The third category includes support systems that play a key role in all production planning and control operations. The last category focuses on the actual workforce and organizational requirements to achieve agile manufacturing.

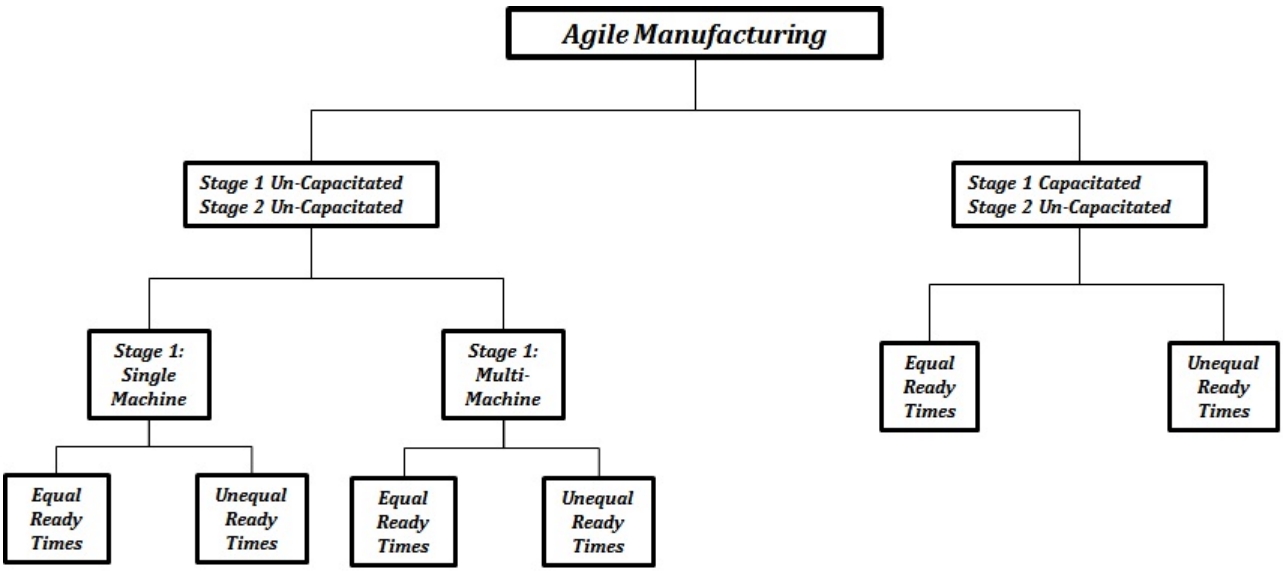
A more recent and extensive survey of the literature of agile manufacturing was done by Sanchez and Nagi [64]. As part of their survey, they classify the work presented on agile manufacturing into nine categories, as follows: product and manufacturing systems, process planning, production planning, scheduling and control, facilities design and location, material handling and storage systems, information systems, supply chain, human factors, and lastly, business practices and processes. Of the nine different areas, the production planning, scheduling and control are the most closely related to our work and our agile manufacturing problem discussed previously. Within this category, two of the five papers present a scheduling algorithm and mathematical formulation, respectively.

As noted above, under agile manufacturing, a firm can focus on several different areas to improve agility. With the move towards agility, the capability of a machine to service different types of jobs increases flexibility of the manufacturing system. The agile

manufacturing system that we consider is a two-stage product assembly system. In the first stage, there is one machine available that processes various kinds of jobs. Each job requires the same processing time. The machine in the first stage can process jobs in batches. In setting up a mathematical formulation, we use modeling techniques equivalent to lot streaming to determine the optimal batch sizes, number of batches and their flow to the second stage. The second stage assembles these jobs into different products. Each product has a unique set of job requirements.

The complexity of this model depends on the different features that make up the problem. Similar to the features of the hybrid flow shop, the features we address for modeling the agile manufacturing problem are capacitated or un-capacitated first stage, and equal or unequal ready times. Capacity in this case is defined as restrictions on the size of the batch a machine can handle. Only the first stage can be capacitated or un-capacitated. The second stage assembles the products that have fixed requirements. The other feature, that is, ready times, also only apply to the first stage. Components arriving to the system at distinct ready times are harder to solve than the case where components arrive at the same time.

Each of the different configurations that we address for the agile manufacturing environment is noted in the chart displayed in Figure 6.6. Next, we present notation and the models for each of these configurations, and describe column generation-based methodologies for their solution and the corresponding optimum seeking mixed-integer formulations. The notation for all the agile manufacturing problem configurations is similar and is listed in the notation section. The notation for the master problems, pricing problems and MIP formulations are listed as follows.



Assumptions
- Two Stages
- Minimizing Total Completion Time
- Second Stage always un-capacitated
- Capacity defined by capacity restrictions on the size of a batch a machine can handle
- Single Lot-multiple components intermingle in first stage for processing
- Components assembled into a product at Stage 2

Assumptions
- Two Stages
- Minimizing Total Completion Time
- Stage 1: Single Machine
- Stage 2 always un-capacitated
- Capacity defined by capacity restrictions on the size of a batch a machine can handle
- Single Lot-multiple components intermingle in first stage for processing
- Components assembled into a product at Stage 2

Figure 6.6: Agile Manufacturing Extensions

Notation for Master Problem

Parameters

S	Total number of jobs
σ	Total number of products
s_l	Number of jobs of product type l, $\forall\ l = 1, ..., \sigma$
R_s	Ready time of job s, $\forall\ s = 1, ..., S$
$C_{st}^{(1)}$	Completion time of job s in iteration t an Stage 1, $\forall\ s = 1, ..., S,\ t = 1, ..., T$
$C_{st}^{(2)}$	Completion time of job s in iteration t an Stage 2 $\forall\ s = 1, ..., S,\ t = 1, ..., T$

Variables

$W_s^{(2)}$ Waiting time of jobs at Stage 2 before starting its processing, $\forall\ s = 1, ..., S$

$$x_t^{(1)} = \begin{cases} 1, & \text{if schedule } t \text{ is selected at Stage 1} \\ 0, & \text{otherwise, } \forall\ t = 1, ..., T. \end{cases}$$

$$x_t^{(2)} = \begin{cases} 1, & \text{if schedule } t \text{ is selected at Stage 2} \\ 0, & \text{otherwise, } \forall\ t = 1, ..., T. \end{cases}$$

Notation for Pricing Problems

Parameters

$P^{(1)}$	Processing time for Stage 1
$B^{(1)}$	Maximum number of sublots per machine for Stage 1
$U^{(1)}$	Maximum sublot size in Stage 1
$L^{(1)}$	Minimum sublot size in Stage 1
$M^{(1)}$	Number of machines in Stage 1
B	Total number of products
Req_{bl}	Required number of jobs of type l for product b, $\forall\ l = 1, ..., \sigma,\ b = 1, ..., B$
D_b	Due date of product b, $\forall\ b = 1, ..., B$
$\gamma^{(1)}$	Dual variable for constraint
$\gamma_l^{(2)}$	Dual variable for constraint
$\mu_s^{(1)}$	Dual variable for constraint

Variables

$$y_{si}^{(1)} = \begin{cases} 1, & \text{if job } s \text{ of type } l \text{ contained in sublot } i \\ 0, & \text{otherwise, } \ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., t \end{cases}$$

$$y_{sb}^{(2)} = \begin{cases} 1, & \text{if job } s \text{ assembled in product } b \\ 0, & \text{otherwise, } \forall\ s \in s_l,\ l = 1, ..., \sigma, \\ & b = 1, ..., B. \end{cases}$$

Notation for MIP Formulation

Variables

$W_s^{(2)}$ Waiting time o jobs at Stage 2 before starting processing, $\forall\ s = 1, ..., S$
$S_{im}^{(1)}$ Size of sublot i on machine m, $\forall\ i = 1, .., B^{(1)},\ m = 1, ..., M^{(1)}$
$S_{lka}^{(2)}$ Size of sublot type l of sublot k on machine a, $\forall\ l = 1, ..., \sigma,\ k = 1, .., B^{(2)}$, $a = 1, ..., M^{(2)}$
$C_{im}^{(1)}$ Completion time of sublot i on machine m
$C_b^{(2)}$ Completion time of product b
$C_s^{(1)}$ Completion time of job s in Stage 1, $\forall\ s = 1, ..., S$
$C_s^{(2)}$ Completion time of job s in Stage 2, $\forall\ s = 1, ..., S$

$$Z_{smi}^{(1)} = \begin{cases} 1, & \text{if job } s \text{ of type } l \text{ contained in sublot } i \text{ on machine } a \\ 0, & \text{otherwise, } \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., j \end{cases}$$

$$Z_{slak}^{(2)} = \begin{cases} 1, & \text{if job } s \text{ of type } l \text{ with contained in sublot } k \text{ with machine } a \\ 0, & \text{otherwise, } \forall\ s \in s_l,\ a = 1, ..., M_l^{(2)}\ l = 1, ..., \sigma, \\ & k = 1, ..., B^{(2)}. \end{cases}$$

$$X_{im}^{(1)} = \begin{cases} 1, & \text{if sublot } i \text{ on machine } m \text{ is formed} \\ 0, & \text{otherwise.} \end{cases}$$

6.2.1 Un-Capacitated Agile Manufacturing

In the set of un-capacitated agile manufacturing problems discussed in this section, the models share the same basic setup with a few alterations. Primarily, we define un-capacitated to reference the case when the first stage has no restriction on the size of a batch the single machine can handle at once. We apply the Decomposition II method presented in Chapter 4 to both model configurations for the un-capacitated models shown in 6.6. Using the Decomposition II method, the completion time values of the first stage constitute a bound on the completion time values of the second stage in order to generate feasible schedules. Using this format, the subproblems can be solved sequentially. The formulations for the two problems shown in Figure 6.6 that have an un-capacitated first stage are presented in this section. One model configuration is a single machine in Stage 1 and the second model

configuration is a one with multiple parallel machines in Stage 1. We present formulations that minimize total completion time using Decomposition II for each model configuration as well as an optimum seeking mixed-integer program formulation.

Stage 1 Single Machine Decomposition II Formulation

Minimize:

$$\sum_{l=1}^{\sigma}\sum_{s\in s_l}\sum_{t=1}^{T}\left((C_{st}^{(1)})x_t^{(1)}+(C_{st}^{(2)})x_t^{(2)}\right) \tag{6.99}$$

Subject to:

$$(\gamma^{(1)}) \quad \sum_{t=1}^{T} x_t^{(1)} = 1. \tag{6.100}$$

$$(\gamma^{(2)}) \quad \sum_{t=1}^{T} x_t^{(2)} = 1. \tag{6.101}$$

Complicating Constraints

$$(\mu_s^{(1)}) \quad \sum_{t=1}^{T}\left((C_{st}^{(2)}-P^{(2)})x_t^{(2)}-C_{st}^{(1)}x_t^{(1)}\right) = w_s^{(2)}, \quad \forall\ s\in s_l,\ l=1,...,\sigma. \tag{6.102}$$

Dual

Maximize:

$$\gamma^{(1)}+\gamma^{(2)} \tag{6.103}$$

Subject to:

$$(x^{(1)}) \quad -\mu_s^{(1)}C_{st}^{(1)}+\gamma^{(1)} \le \sum_{s=1}^{S} C_{st}^{(1)}. \tag{6.104}$$

$$(x^{(2)}) \quad \sum_{l=1}^{\sigma} \sum_{s \in s_l} (\mu_s^{(1)}) C_{st}^{(2)} + \gamma^{(2)} \leq \sum_{l=1}^{\sigma} \sum_{s \in s_l} C_{st}^{(2)}. \tag{6.105}$$

$$\mu_s^{(1)} \ \gamma^{(1)}, \ \gamma^{(2)} \text{ unrestricted.} \tag{6.106}$$

Pricing Problems:

ZPP1:

Minimize:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} \sum_{i=1}^{j} \left((1 + \mu_s^{(1)}) C_{si}^{(1)} \right) - \gamma^{(1)} \tag{6.107}$$

Subject to:

$$\begin{gathered} \sum_{i=1}^{j} y_{si}^{(1)} = 1, \\ \forall \ s \in s_l, \ l = 1, ..., \sigma. \end{gathered} \tag{6.108}$$

$$\begin{gathered} C_{si}^{(1)} \leq (P^{(1)} + R_s) y_{si}^{(1)}, \\ \forall \ s \in s_l, \ l = 1, ..., \sigma, \ i = 1, ..., j. \end{gathered} \tag{6.109}$$

$$\begin{gathered} C_{si}^{(1)} - (P^{(1)} + R_s) y_{si}^{(1)} \geq 0, \\ \forall \ i = 1, ..., j, \ s \in s_l, \ l = 1, ..., \sigma. \end{gathered} \tag{6.110}$$

$$\begin{gathered} C_{s'i}^{(1)} - C_{si-1}^{(1)} - P^{(1)} + G(1 - y_{s'i}^{(1)}) \geq 0, \\ \forall \ i = 2, ..., j, \ s' \in s_l, \ s \in s_l, \ l = 1, ..., \sigma, \ s' \neq s. \end{gathered} \tag{6.111}$$

$$\begin{gathered} C_{s'i}^{(1)} - C_{si}^{(1)} + G(1 - y_{s'i}^{(1)}) + G(1 - y_{si}^{(1)}) \geq 0, \\ \forall \ i = 1, ..., j, \ s' \in s_l, \ s \in s_l, \ l = 1, ..., \sigma, \ s' \neq s. \end{gathered} \tag{6.112}$$

ZPP2:

Minimize

$$\sum_{l=1}^{\sigma}\sum_{s\in s_l}\left(\sum_{b=1}^{B}C_{sb}^{(2)}-\mu_s^{(1)}(C_{sb}^{(2)}-P_b^{(2)})\right)-\gamma^{(2)} \tag{6.113}$$

Subject to:

$$\begin{gathered}\sum_{b=1}^{B}y_{sb}^{(2)}=1,\\ \forall\ s\in s_l,\ l=1,...,\sigma.\end{gathered} \tag{6.114}$$

$$\begin{gathered}\sum_{s\in s_l}y_{sb}^{(2)}=Req_{lb},\\ \forall\ l=1,...,\sigma,\ b=1,...,B.\end{gathered} \tag{6.115}$$

Product Feasibility:

$$\begin{gathered}C_{sb}^{(2)}-(P^{(2)}+C_s^{(1)})+G(1-y_s^{(1)})+G(1-y_{sb}^{(2)})\geq 0\\ \forall\ s\in s_l,\ l=1,...,\sigma,\ b=1,...,B.\end{gathered} \tag{6.116}$$

Product Completion Equality:

$$\begin{gathered}C_{s'b}^{(2)}-C_{sb}^{(2)}+G(1-y_{s'b}^{(2)})+G(1-y_{sb}^{(2)})\geq 0,\\ \forall\ \ s'\in s_l,\ s\in s_l\ s'\neq s,\ l=1,...,\sigma,\ b=1,...,B.\end{gathered} \tag{6.117}$$

Product Completion Times:

$$\begin{gathered}C_{sb}^{(2)}\leq D_b(y_{sb}^{(2)}),\\ \forall\ s\in s_l,\ b=1,...,B,\ l=1,...,\sigma.\end{gathered} \tag{6.118}$$

Stage 1 Single Machine MIP Formulation

Minimize:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} \left(CS_s^{(1)} + CS_s^{(2)} \right)$$

Stage 1

Sublot Assignment:

$$\sum_{i=1}^{B^{(1)}} Z_{sli}^{(1)} = 1, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.119}$$

Wait Time:

$$0 \leq C_i^{(1)} - P^{(1)} - R - G(1 - Z_{sli}^{(1)}) \leq W_s^{(1)} \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., j. \tag{6.120}$$

Sublot Order:

$$C_1^{(1)} - P^{(1)} \geq 0, \tag{6.121}$$

$$C_{(i+1)}^{(1)} - C_i^{(1)} - P^{(1)} \geq 0, \quad \forall\ i = 1, ..., j. \tag{6.122}$$

Stage 1 Completion Times:

$$C_1^{(1)} - P^{(1)} - R_s + G(1 - Z_{sl1}^{(1)}) \geq 0, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.123}$$

$$C_{im}^{(1)} - P^{(1)} - R_s + G(1 - Z_{sli}^{(1)}) \geq 0, \quad \forall\ i = 2, ..., j-1,\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.124}$$

Stage 2

$$\sum_{b=1}^{B} Z_{slb}^{(2)} = 1, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.125}$$

$$\sum_{s \in s_l} Z_{slb}^{(2)} = Req_{bl}, \quad \forall\ l = 1, ..., \sigma,\ b = 1, ..., B. \tag{6.126}$$

Product Completion Times:

$$C_b^{(2)} - (P^{(2)} + C_i^{(1)}) + G(1 - Z_{si}^{(1)}) + G(1 - Z_{sb}^{(2)}) \geq 0, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ b = 1, ..., B,\ i = 1, ..., B^{(1)}. \tag{6.127}$$

Job Completion Times:

$$C_b^{(2)} - G(1 - Z_{slb}^{(2)}) \leq CS_{slb}^{(2)}, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ b = 1, ..., B. \tag{6.128}$$

Product Due Date:

$$C_b^{(2)} - (P^{(2)} + C_i^{(1)}) + G(1 - Z_{si}^{(1)}) + G(1 - Z_{slb}^{(2)}) \leq D_b, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ b = 1, ..., B,\ i = 1, ..., B^{(1)}. \tag{6.129}$$

Equal Ready Times

For the equal ready time model configuration, the same formulation as presented above is applicable. The equal ready times are accounted for through the parameter settings for

ready times for the jobs arriving to the system. With all the jobs arriving to the system at equivalent times, the model complexity is much less than that for the unequal ready times, which is presented in the next section.

Results

The performances of the Decomposition II approach and the direct solution by CPLEX for the equal ready time case for the un-capacitated single-machine first stage agile manufacturing environment are shown in Figure 6.7. For each approach, we ran ten scenarios for various model variations. We varied the model by the number of job types and the number of jobs. For each model variation we ran ten scenarios representing ten sets of processing times. As we have observed in previous examples, the behavior shown in Figure 6.7 is very similar to Figure 6.2. The average required CPU times required for the Decomposition II approach and CPLEX solving the problem directly are fairly similar for the smaller model variation scenarios. The spike-like behavior occurs for the same model variation (2T2M15J) when the number of jobs increases but for few job types.

Unequal Ready Times

In the unequal ready time model, the jobs arrive at the system at distinct times, which complicates the determination of the sublot completion times in Stage 1. To test our solution method (Decomposition II) for this model configuration, we used several scenarios by varying parameters and compared its performance with that of the MIP formulation solved using CPLEX. The model variations include changes to the number of job types and the number of jobs. Each model variation is run over a set of ten different ready times. The parameters for the model are listed in Table 6.3. The results of these performances for Stage 1 single machine model configuration are presented in the following section.

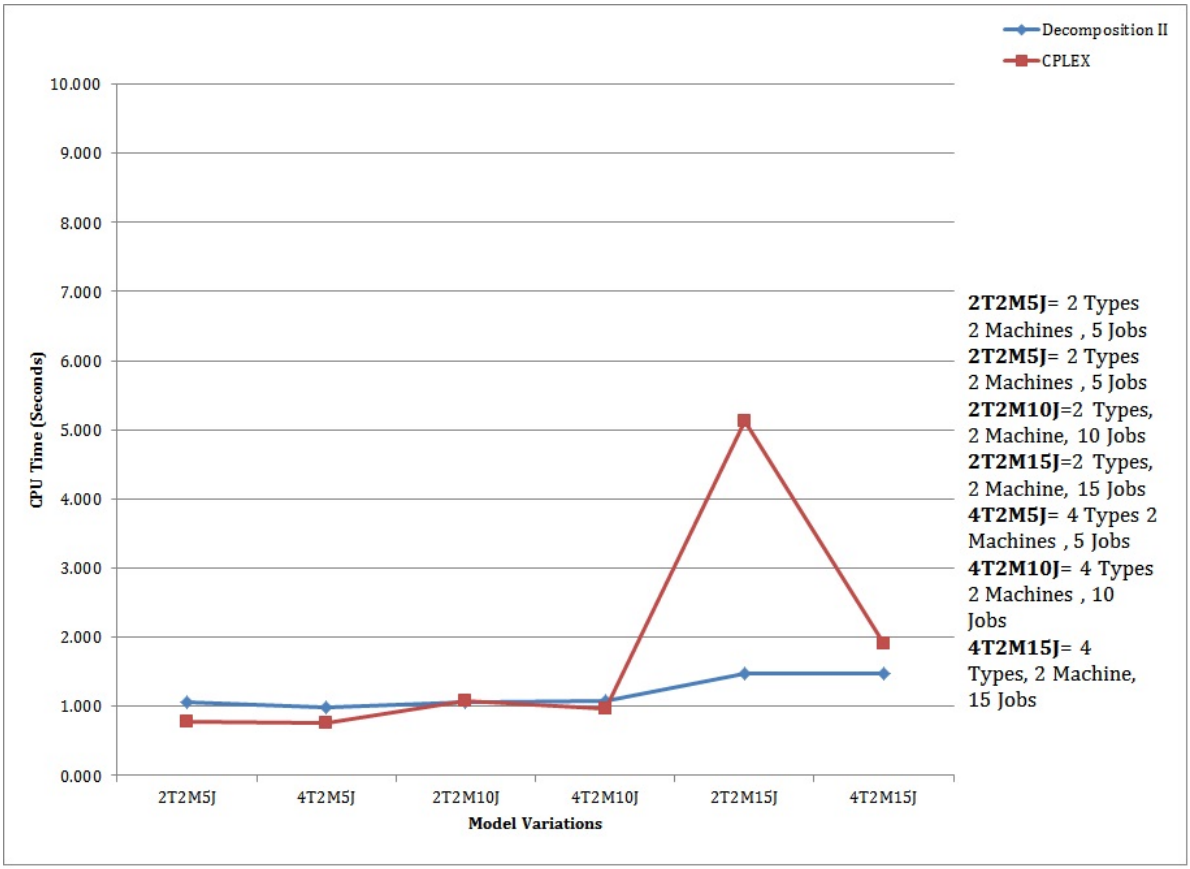


Figure 6.7: Un-Capacitated, Equal Ready Times, Single Machine Stage 1 Agile Manufacturing: Average performance values for solutions obtained by Decomposition II and directly using CPLEX

Results

The graph in Figure 6.8 displays the average of the performance results for the Stage 1 Single Machine Model Configuration under different variations. Consistent with previous results, the solutions obtained directly using CPLEX and those obtained by the Decomposition II method, experience similar performance results for small-sized problem instances. As the number of jobs increases, the average required by solutions obtained directly using CPLEX also increases significantly for the 15-job problem instance. Likewise, the average CPU time required by the Decomposition II method stays relatively the same for each model variation. The accuracy of the Decomposition II increases with larger-sized problem instance; however, the gap percentage between Decomposition II and the optimal solution obtained by CPLEX

is less than 1% for every model variation.

Table 6.3: Agile Manufacturing Parameters

Parameter	Value
$B^{(1)}$	3
$P^{(1)}$	2
Total Products	3
$P^{(2)}$	2, 3, 2

Stage 1 Multiple Machine Model Configuration

The formulation for the multiple machines in Stage 1 is very similar to the formulation with a single machine in Stage 1. The main difference is essentially an additional index to the variables in the first stage to account for the different number of machines that a batch of jobs can be processed on. The additions in both the Decomposition II and MIP formulations from the single machine case in the first stage include an indicator variable for when a sublot on a machine is formed. The Decomposition II formulation is presented first followed by the MIP formulation.

Decomposition II Formulation

Minimize:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} \sum_{t=1}^{T} \left((C_{st}^{(1)}) x_t^{(1)} + (C_{st}^{(2)}) x_t^{(2)} \right) \tag{6.130}$$

Subject to:

$$(\gamma^{(1)}) \quad \sum_{t=1}^{T} x_t^{(1)} = 1. \tag{6.131}$$

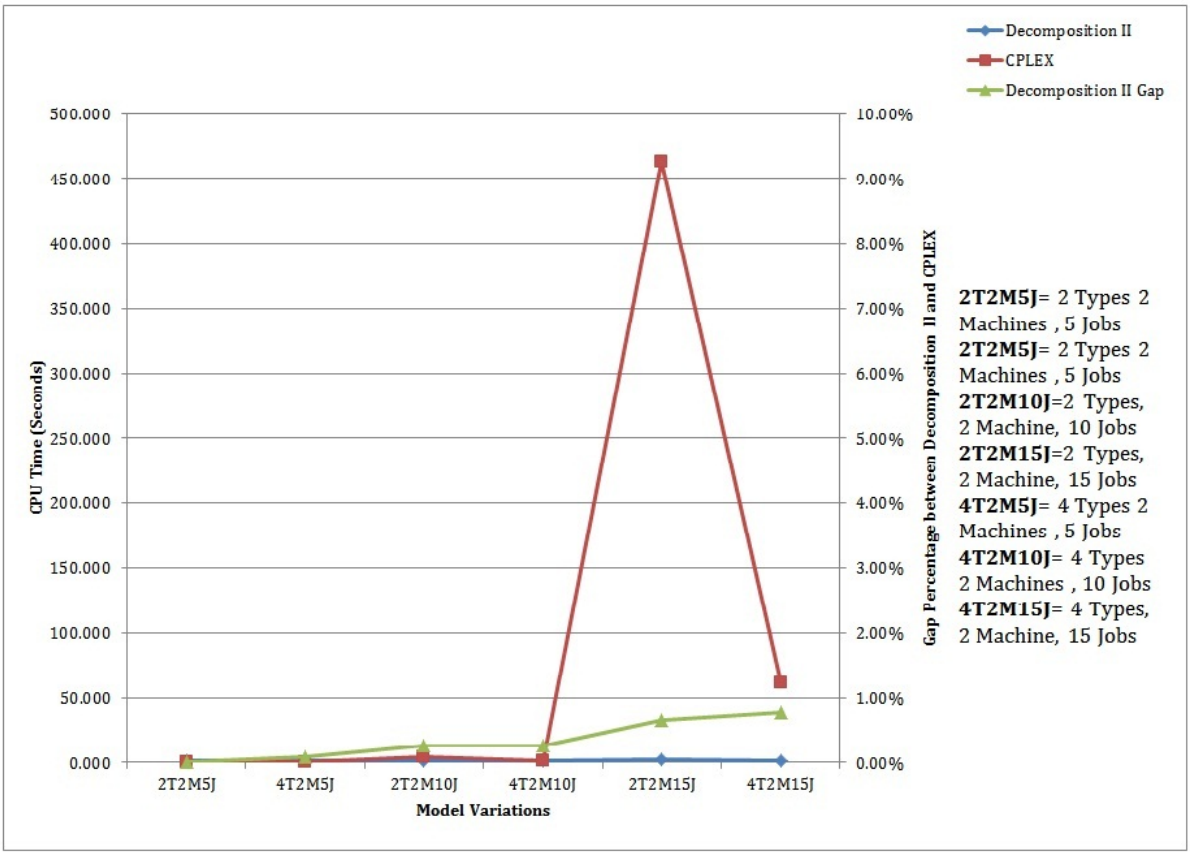


Figure 6.8: Un-Capacitated Single Machine Stage 1 Agile Manufacturing: Average performance values for solutions obtained by Decomposition II and directly using CPLEX

$$(\gamma^{(2)}) \quad \sum_{t=1}^{T} x_t^{(2)} = 1. \tag{6.132}$$

Complicating Constraints

$$(\mu_s^{(1)}) \quad \sum_{t=1}^{T} \left((C_{st}^{(2)} - P^{(2)}) x_t^{(2)} - C_{st}^{(1)} x_t^{(1)} \right) = w_s^{(2)}, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.133}$$

Dual

Maximize:

$$\gamma^{(1)} + \gamma^{(2)} \tag{6.134}$$

Subject to:

$$(x^{(1)}) \quad -\mu_s^{(1)} C_{st}^{(1)} + \gamma^{(1)} \leq \sum_{s=1}^{S} C_{st}^{(1)}. \tag{6.135}$$

$$(x^{(2)}) \quad \sum_{l=1}^{\sigma} \sum_{s \in s_l} (\mu_s^{(1)})(C_{st}^{(2)} - P^{(2)}) + \gamma^{(2)} \leq \sum_{l=1}^{\sigma} \sum_{s \in s_l} C_{st}^{(2)}. \tag{6.136}$$

$$\mu_s^{(1)} \ \gamma^{(1)}, \ \gamma^{(2)} \text{ unrestricted.} \tag{6.137}$$

Pricing Problems:

ZPP1:

Minimize:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} \sum_{m=1}^{M^{(1)}} \sum_{i=1}^{B^{(1)}} \left((1 + \mu_s^{(1)}) C_{smi}^{(1)} \right) - \gamma^{(1)} \tag{6.138}$$

Subject to:

$$\sum_{m=1}^{M^{(1)}} \sum_{i=1}^{B^{(1)}} y_{smi}^{(1)} = 1, \quad \forall \ s \in s_l, \ l = 1, ..., \sigma. \tag{6.139}$$

$$C_{smi}^{(1)} \leq (P^{(1)} + R_s) y_{smi}^{(1)}, \quad \forall \ s \in s_l, \ l = 1, ..., \sigma, \ m = 1, ..., M^{(1)}, \ i = 1, ..., B^{(1)}. \tag{6.140}$$

$$C_{smi}^{(1)} - (P^{(1)} + R_s) y_{smi}^{(1)} \geq 0, \quad \forall \ m = 1, ..., M^{(1)}, \ i = 1, ..., j, \ s \in s_l, \ l = 1, ..., \sigma. \tag{6.141}$$

$$C^{(1)}_{s'mi} - C^{(1)}_{smi-1} - P^{(1)} + G(1 - y^{(1)}_{s'mi}) \geq 0,$$
$$\forall\ m = 1, ..., M^{(1)},\ i = 2, ..., B^{(1)},$$
$$s' \in s_l,\ s \in s_l,\ l = 1, ..., \sigma,\ s' \neq s. \tag{6.142}$$

$$C^{(1)}_{s'mi} - C^{(1)}_{smi} + G(1 - y^{(1)}_{s'mi}) + G(1 - y^{(1)}_{smi}) \geq 0,$$
$$\forall\ m = 1, ..., M^{(1)},\ i = 1, ..., B^{(1)},$$
$$s' \in s_l,\ s \in s_l,\ l = 1, ..., \sigma\ s' \neq s. \tag{6.143}$$

ZPP2:

Minimize

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} \left(\sum_{b=1}^{B} C^{(2)}_{sb} - \mu^{(1)}_s (C^{(2)}_{sb} - P^{(2)}_b) \right) - \gamma^{(2)} \tag{6.144}$$

Subject to:

$$\sum_{b=1}^{B} y^{(2)}_{sb} = 1,$$
$$\forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.145}$$

$$\sum_{s \in s_l} y^{(2)}_{sb} = Req_{lb},$$
$$\forall\ l = 1, ..., \sigma,\ b = 1, ..., B. \tag{6.146}$$

Product Feasibility:

$$C^{(2)}_{sb} - (P^{(2)} + C^{(1)}_s) + G(1 - y^{(1)}_s) + G(1 - y^{(2)}_{sb}) \geq 0,$$
$$\forall\ s \in s_l,\ l = 1, ..., \sigma,\ b = 1, ..., B. \tag{6.147}$$

Product Completion Equality:

$$C_{s'b}^{(2)} - C_{sb}^{(2)} + G(1 - y_{s'b}^{(2)}) + G(1 - y_{sb}^{(2)}) \geq 0, \\ \forall \ \ s' \in s_l, \ s \in s_l \ s' \neq s, \ l = 1, ..., \sigma, \ b = 1, ..., B. \tag{6.148}$$

Product Completion Times:

$$C_{sb}^{(2)} \leq D_b(y_{sb}^{(2)}), \\ \forall \ s \in s_l, \ b = 1, ..., B, \ l = 1, ..., \sigma. \tag{6.149}$$

MIP Formulation

Minimize:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} \left(CS_s^{(1)} + CS_s^{(2)} \right)$$

Stage 1

Sublot Assignment:

$$\sum_{m=1}^{M^{(1)}} \sum_{i=1}^{B^{(1)}} Z_{slim}^{(1)} = 1, \\ \forall \ s \in s_l, \ l = 1, ..., \sigma. \tag{6.150}$$

Sublot Formation Indicator Variable:

$$\sum_{m=1}^{M^{(1)}} X_{im}^{(1)} \leq 1, \\ \forall \ i = 1, ..., B^{(1)}. \tag{6.151}$$

Sublot Indicator Bound:

$$G(X_{im}^{(1)}) \geq S_{im}^{(1)}, \\ \forall \ i = 1, ..., B^{(1)} \ m = 1, ..., M^{(1)}. \tag{6.152}$$

Wait Time:

$$\begin{gathered} 0 \leq C_{im}^{(1)} - P^{(1)} - R - G(1 - Z_{slim}^{(1)}) \leq W_s^{(1)} \\ \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., B^{(1)},\ m = 1, ..., M^{(1)}. \end{gathered} \tag{6.153}$$

Sublot Order:

$$C_{11}^{(1)} - P^{(1)} \geq 0, \tag{6.154}$$

$$\begin{gathered} C_{(i+1)m}^{(1)} - C_{im}^{(1)} - P^{(1)} \geq 0, \\ \forall\ i = 1, ..., B^{(1)},\ m = 1, ..., M^{(1)}. \end{gathered} \tag{6.155}$$

Completion Times:

$$\begin{gathered} C_{11}^{(1)} - P^{(1)} - R_s + G(1 - Z_{sl11}^{(1)}) \geq 0, \\ \forall\ s \in s_l,\ l = 1, ..., \sigma. \end{gathered} \tag{6.156}$$

$$\begin{gathered} C_{im}^{(1)} - P^{(1)} - R_s + G(1 - Z_{slim}^{(1)}) \geq 0, \\ \forall\ i = 2, ..., B^{(1)} - 1,\ m = 1, ..., M^{(1)},\ s \in s_l,\ l = 1, ..., \sigma. \end{gathered} \tag{6.157}$$

Stage 2

$$\begin{gathered} \sum_{b=1}^{B} Z_{slb}^{(2)} = 1, \\ \forall\ s \in s_l,\ l = 1, ..., \sigma. \end{gathered} \tag{6.158}$$

$$\begin{gathered} \sum_{s \in s_l} Z_{slb}^{(2)} = Req_{bl}, \\ \forall\ l = 1, ..., \sigma,\ b = 1, ..., B. \end{gathered} \tag{6.159}$$

Product Completion Times:

$$C_b^{(2)} - (P^{(2)} + C_i^{(1)}) + G(1 - Z_{si}^{(1)}) + G(1 - Z_{sb}^{(2)}) \geq 0, \\ \forall\ s \in s_l,\ l = 1, ..., \sigma,\ b = 1, ..., B,\ i = 1, ..., B^{(1)}. \tag{6.160}$$

Job Completion Times:

$$C_b^{(2)} - G(1 - Z_{slb}^{(2)}) \leq CS_{slb}^{(2)}, \\ \forall\ s \in s_l,\ l = 1, ..., \sigma,\ b = 1, ..., B. \tag{6.161}$$

Product Due Date:

$$C_b^{(2)} - (P^{(2)} + C_i^{(1)}) + G(1 - Z_{si}^{(1)}) + G(1 - Z_{slb}^{(2)}) \leq D_b, \\ \forall\ s \in s_l,\ l = 1, ..., \sigma,\ b = 1, ..., B,\ i = 1, ..., B^{(1)}. \tag{6.162}$$

Equal Ready Times

The model configuration for equal ready times is identical to the formulation shown for Decomposition II and the MIP. The equal ready times are thus accounted for through the ready time parameters. For this particular model configuration, the ready times for the job are set equal. With equal ready times and an un-capacitated Stage 1, the model is less complicated than the unequal ready time model configuration discussed in the next section.

Results

We present the performance results of the Decomposition II approach and direct solution obtained using CPLEX in Figure 6.9 for the un-capacitated, multi-machine first stage agile manufacturing environment. For the six model variations, we notice a similar behavior as we have observed previously. The average CPU times required by the Decomposition II method is fairly steady for every model variation. On the other hand, the solution obtained directly

using CPLEX required on average about the same CPU times for smaller-sized scenarios but increased dramatically for the 2T2M15J variation.

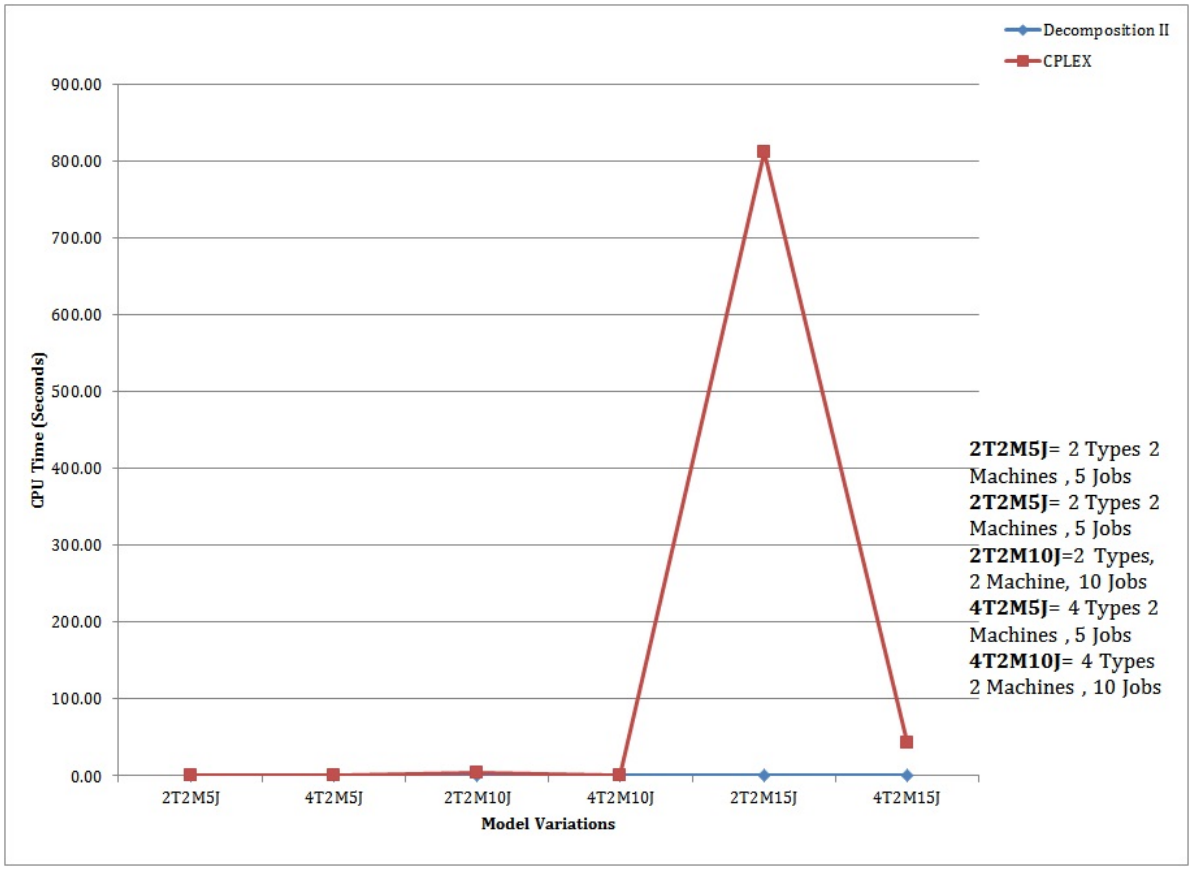


Figure 6.9: Un-Capacitated, Equal Ready Times, Multi-Machine Stage 1 Agile Manufacturing: Average performance values for solutions obtained by Decomposition II and directly using CPLEX

Unequal Ready Times

The model configuration for jobs arriving at distinct times is much more complicated for the determination of sublot completion times. With multiple machines in the first stage that are un-capacitated, this model configuration has more flexibility to process multiple sublots simultaneously, given the number of machines. To test the performance of the Decomposition II method compared with that of direct solution by CPLEX, we perform several test runs for different model variations. These include changes to the number of job types and the

number of jobs. For each model variation, ten runs were performed for a set of ten different ready times. The only change to parameters from those listed in Table 6.3 is the addition of the number of machines allowed at Stage 1. We use $M^{(1)} = 2$ for the test runs. The results are presented in the next section.

Results

The results for performance comparison of solutions obtained for the Stage 1 Multiple Machine Configuration by the Decomposition II method and direct solution by CPLEX are presented in the graph shown in Figure 6.10. With the additional machine in Stage 1, the complexity of the model is increased. The test runs for the large-sized problem instances of 15 jobs were too large for CPLEX to handle. We observe that for small-sized problem instances, both solution approaches require very little CPU time on average. For the medium-sized problem instance of 10 jobs, we see a significant increase in the average CPU time required. The Decomposition II method performance remains relatively steady with respect to the average required CPU time. The accuracy of the Decomposition II method stays under 1%.

6.2.2 Capacitated Agile Manufacturing

In the two-stage agile manufacturing problem with a capacitated first stage, we formulate the case of a single machine with restricted sublot sizes. The formulation restricts the size of the batch the machine at the first stage can handle by allowing a range of values the batch size could take. The components that arrive to the system have distinct ready times, and therefore, the batch completion times are calculated with respect to the ready times of each component that is assigned to a batch. We present a formulation using the Decomposition II method presented in Chapter 4 to solve the problem sequentially. In order to compare the performance of the Decomposition II method, we present a mixed-integer program formulation. The formulation of the model is as follows:

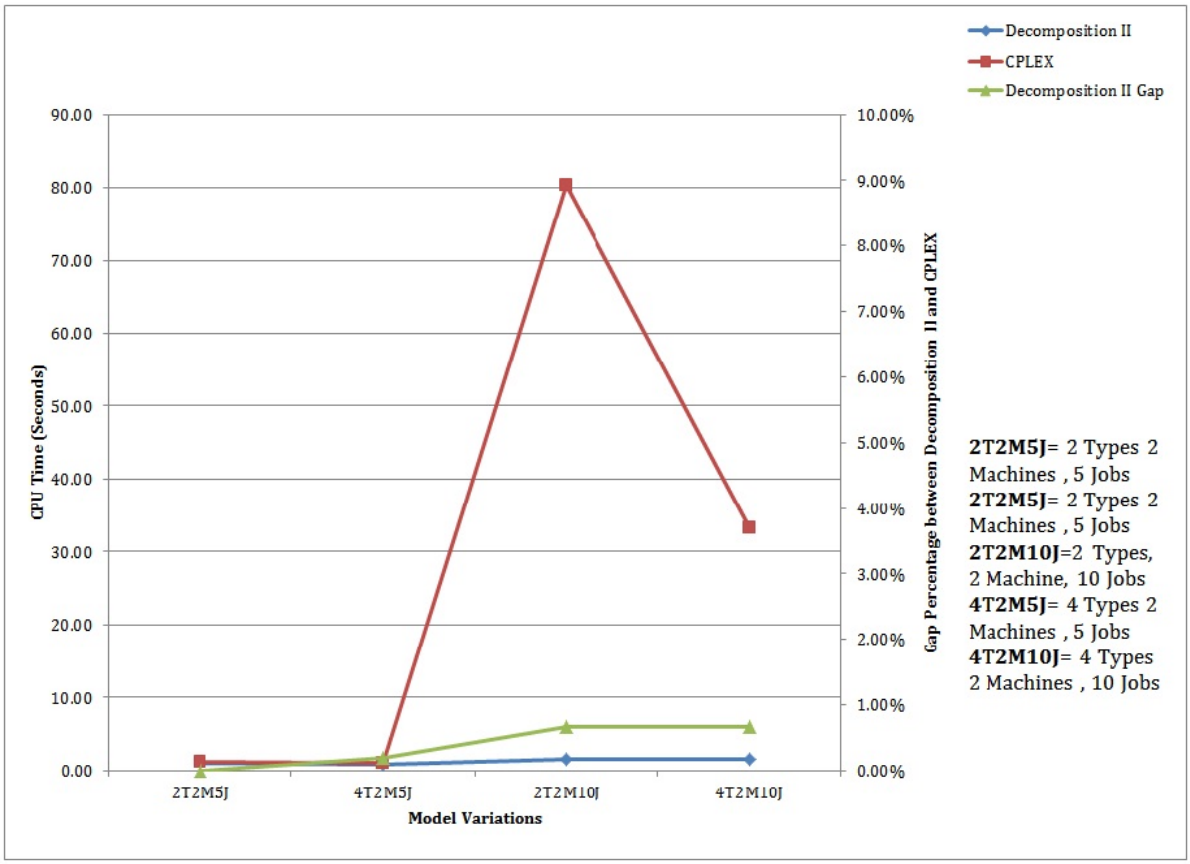


Figure 6.10: Un-Capacitated Multiple Machine Stage 1 Agile Manufacturing: Average performance values for solutions obtained by Decomposition II and directly using CPLEX

Stage 1 Single Machine Decomposition II Formulation

Minimize:

$$\sum_{l=1}^{\sigma}\sum_{s\in s_l}\sum_{t=1}^{T}\left((C_{st}^{(1)})x_t^{(1)}+(C_{st}^{(2)})x_t^{(2)}\right) \tag{6.163}$$

Subject to:

$$(\gamma^{(1)}) \quad \sum_{t=1}^{T} x_t^{(1)} = 1. \tag{6.164}$$

$$(\gamma^{(2)}) \quad \sum_{t=1}^{T} x_t^{(2)} = 1. \tag{6.165}$$

$$(\mu_s^{(1)}) \quad \sum_{t=1}^{T} \left((C_{st}^{(2)} - P^{(2)}) x_t^{(2)} - C_{st}^{(1)} x_t^{(1)} \right) = w_s^{(2)}, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.166}$$

Dual

Maximize:

$$\gamma^{(1)} + \gamma^{(2)} \tag{6.167}$$

Subject to:

$$(x^{(1)}) \quad -\mu_s^{(1)} C_{st}^{(1)} + \gamma^{(1)} \leq \sum_{s=1}^{S} C_{st}^{(1)}. \tag{6.168}$$

$$(x^{(2)}) \quad \sum_{l=1}^{\sigma} \sum_{s \in s_l} (\mu_s^{(1)}) C_{st}^{(2)} + \gamma^{(2)} \leq \sum_{l=1}^{\sigma} \sum_{s \in s_l} C_{st}^{(2)}. \tag{6.169}$$

$$\mu_s^{(1)}\ \gamma^{(1)},\ \gamma^{(2)} \text{ unrestricted.} \tag{6.170}$$

Pricing Problems:

ZPP1:

Minimize:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} \sum_{i=1}^{j} \left((1 + \mu_s^{(1)}) C_{si}^{(1)} \right) - \gamma^{(1)} \tag{6.171}$$

Subject to:

$$\sum_{i=1}^{j} y_{si}^{(1)} = 1, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.172}$$

Minimum Size:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} y_{si}^{(1)} \geq L^{(1)}, \quad \forall\ i = 1, ..., j. \tag{6.173}$$

Maximum Size:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} y_{si}^{(1)} \leq U^{(1)}, \quad \forall\ i = 1, ..., j. \tag{6.174}$$

$$C_{si}^{(1)} \leq (P^{(1)} + R_s) y_{si}^{(1)}, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., j. \tag{6.175}$$

$$C_{si}^{(1)} - (P^{(1)} + R_s) y_{si}^{(1)} \geq 0, \quad \forall\ i = 1, ..., j,\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.176}$$

$$C_{s'i}^{(1)} - C_{si-1}^{(1)} - P^{(1)} + G(1 - y_{s'i}^{(1)}) \geq 0, \quad \forall\ i = 2, ..., j,\ s' \in s_l,\ s \in s_l,\ l = 1, ..., \sigma,\ s' \neq s. \tag{6.177}$$

$$C_{s'i}^{(1)} - C_{si}^{(1)} + G(1 - y_{s'i}^{(1)}) + G(1 - y_{si}^{(1)}) \geq 0, \quad \forall\ i = 1, ..., j,\ s' \in s_l,\ s \in s_l,\ l = 1, ..., \sigma,\ s' \neq s. \tag{6.178}$$

ZPP2:

Minimize

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} \left(\sum_{b=1}^{B} C_{sb}^{(2)} - \mu_s^{(1)} (C_{sb}^{(2)} - P_b^{(2)}) \right) - \gamma^{(2)} \tag{6.179}$$

Subject to:

$$\sum_{b=1}^{B} y_{sb}^{(2)} = 1, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.180}$$

$$\sum_{s \in s_l} y_{sb}^{(2)} = Req_{lb}, \quad \forall\ l = 1, ..., \sigma,\ b = 1, ..., B. \tag{6.181}$$

Product Feasibility:

$$C_{sb}^{(2)} - (P^{(2)} + C_s^{(1)}) + G(1 - y_s^{(1)}) + G(1 - y_{sb}^{(2)}) \geq 0 \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ b = 1, ..., B. \tag{6.182}$$

Product Completion Equality:

$$C_{s'b}^{(2)} - C_{sb}^{(2)} + G(1 - y_{s'b}^{(2)}) + G(1 - y_{sb}^{(2)}) \geq 0, \quad \forall\ \ s' \in s_l,\ s \in s_l\ s' \neq s,\ l = 1, ..., \sigma,\ b = 1, ..., B. \tag{6.183}$$

Product Completion Times:

$$C_{sb}^{(2)} \leq D_b(y_{sb}^{(2)}), \quad \forall\ s \in s_l,\ b = 1, ..., B,\ l = 1, ..., \sigma. \tag{6.184}$$

Stage 1 Single Machine MIP Formulation

Minimize:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} \left(CS_s^{(1)} + CS_s^{(2)} \right) \tag{6.185}$$

Stage 1

Sublot Assignment:

$$\sum_{i=1}^{B^{(1)}} Z_{sli}^{(1)} = 1, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.186}$$

Sublot Minimum:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} Z_{sli}^{(1)} \geq L^{(1)}, \quad \forall\ i = 1, .., B^{(1)}. \tag{6.187}$$

Sublot Maximum:

$$\sum_{l=1}^{\sigma} \sum_{s \in s_l} Z_{sli}^{(1)} \leq U^{(1)}, \quad \forall\ i = 1, .., j. \tag{6.188}$$

Wait Time:

$$0 \leq C_i^{(1)} - P^{(1)} - R - G(1 - Z_{sli}^{(1)}) \leq W_s^{(1)} \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ i = 1, ..., j. \tag{6.189}$$

Sublot Order:

$$C_1^{(1)} - P^{(1)} \geq 0, \tag{6.190}$$

$$C_{(i+1)}^{(1)} - C_i^{(1)} - P^{(1)} \geq 0, \quad \forall\ i = 1, ..., j. \tag{6.191}$$

Completion Times:

$$C_1^{(1)} - P^{(1)} - R_s + G(1 - Z_{sl1}^{(1)}) \geq 0, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.192}$$

$$C_{im}^{(1)} - P^{(1)} - R_s + G(1 - Z_{sli}^{(1)}) \geq 0, \quad \forall\ i = 2, ..., j-1,\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.193}$$

Stage 2

$$\sum_{b=1}^{B} Z_{slb}^{(2)} = 1, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma. \tag{6.194}$$

$$\sum_{s \in s_l} Z_{slb}^{(2)} = Req_{bl}, \quad \forall\ l = 1, ..., \sigma,\ b = 1, ..., B. \tag{6.195}$$

Product Completion Times:

$$C_b^{(2)} - (P^{(2)} + C_i^{(1)}) + G(1 - Z_{si}^{(1)}) + G(1 - Z_{sb}^{(2)}) \geq 0, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ b = 1, ..., B,\ i = 1, ..., B^{(1)}. \tag{6.196}$$

Job Completion Times:

$$C_b^{(2)} - G(1 - Z_{slb}^{(2)}) \leq CS_{slb}^{(2)}, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ b = 1, ..., B. \tag{6.197}$$

Product Due Date:

$$C_b^{(2)} - (P^{(2)} + C_i^{(1)}) + G(1 - Z_{si}^{(1)}) + G(1 - Z_{slb}^{(2)}) \leq D_b, \quad \forall\ s \in s_l,\ l = 1, ..., \sigma,\ b = 1, ..., B,\ i = 1, ..., B^{(1)}. \tag{6.198}$$

Equal Ready Time

The formulation presented above using the Decomposition II method and the MIP formulation are applicable to the equal ready time model configuration. The ready times are set equal as the parameters for this particular configuration. With equal ready times present in a capacitated single machine at Stage 1 the completions times are less complicated to determine than the unequal ready time case. The number of sublots formed, and thus, their completion times are dependent on the restriction that dictate the minimum and maximum size a sublot can form.

Results

The performance results of the Decomposition II approach and direct solution by CPLEX of the equal ready time case for the capacitated agile manufacturing environment with a single machine in the first stage is presented in Figure 6.11. The average CPU time requirements for the two solution cases are greater than the un-capacitated case. These results are expected given its increased complexity. The similar behavior we have observed thus far is still prevalent in Figure 6.11 if not clearer given the greater CPU time averages.

Unequal Ready Time

For unequal ready times, we compare the performance of the two solution methods presented for the capacitated Stage 1 single-machine model configuration. This model configuration using unequal ready times is more complicated than the equal ready time case. The model must decide completion times that are dependent not only on the restrictions imposed on sublot sizes, but also on the different times at which the jobs arrive at the system, while minimizing total completion time. To test the performance of the Decomposition II method, we run several tests for different model variations that includes changes to the number of

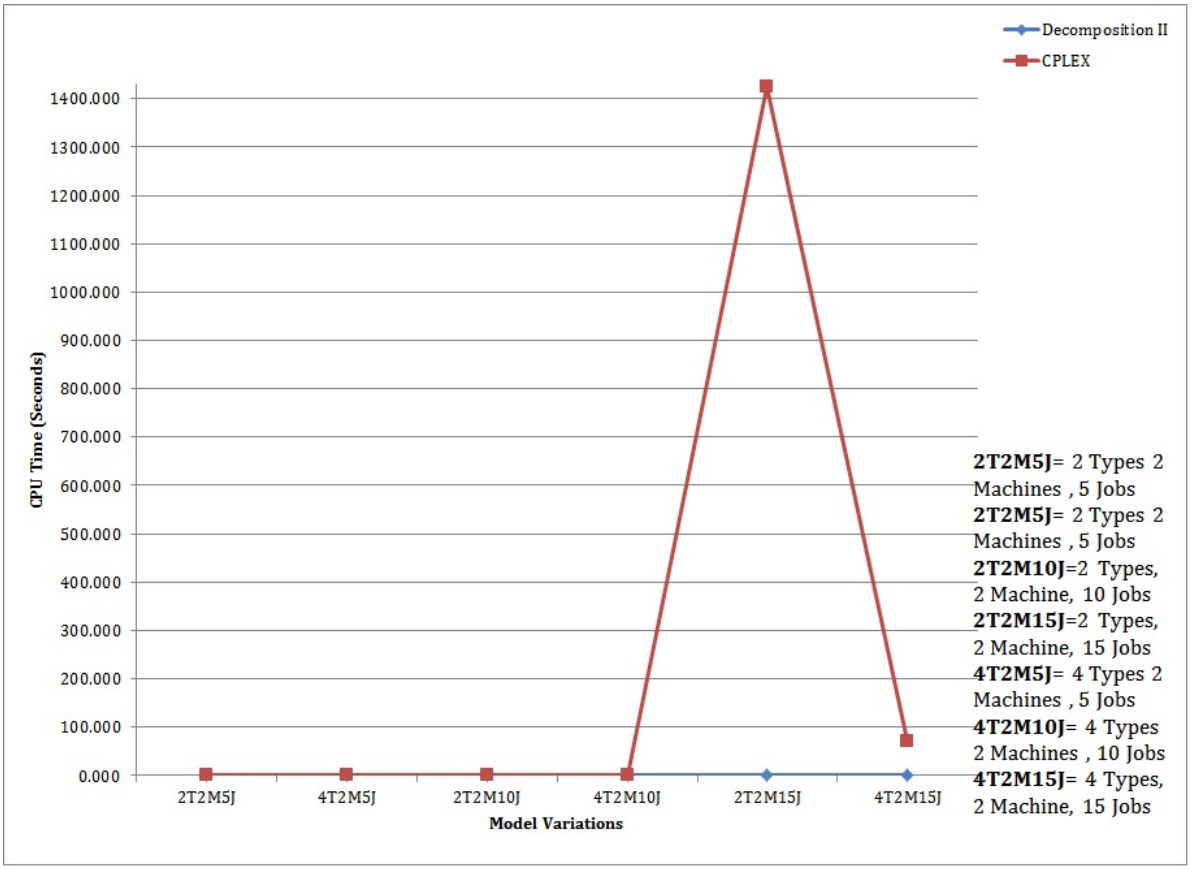


Figure 6.11: Capacitated, Equal Ready Times, Single Machine Stage 1 Agile Manufacturing: Average performance values for solutions obtained by Decomposition II and directly using CPLEX

job types and the total number of jobs. The parameters used for the test runs are the same as those in Table 6.3 with additions listed in Table 6.4. For each model variation, ten runs are performed for ten different sets of unequal ready times.

Table 6.4: Capacitated Agile Manufacturing Parameters

Parameter	Value
$L^{(1)}$	1
$U^{(1)}$	5

Results

The performance comparison of the two solution methods are presented in the graph in Figure 6.12. For the small and medium-sized problem instances, the average time required by both the Decomposition II method and solutions obtained directly using CPLEX is roughly the same. For the large-sized 15 job model variations, the solution obtained directly using CPLEX require on average significantly more CPU time. For the increased number of job types for large-sized problem instance, the solution obtained directly using CPLEX requires less CPU time on average than the model variation with fewer job types for a fixed number of jobs. Unlike the performance of solutions obtained directly using CPLEX, the Decomposition II method required on average about the same CPU time. The accuracy of the Decomposition II method is depicted by a gap of 0.25% from the optimal solution value.

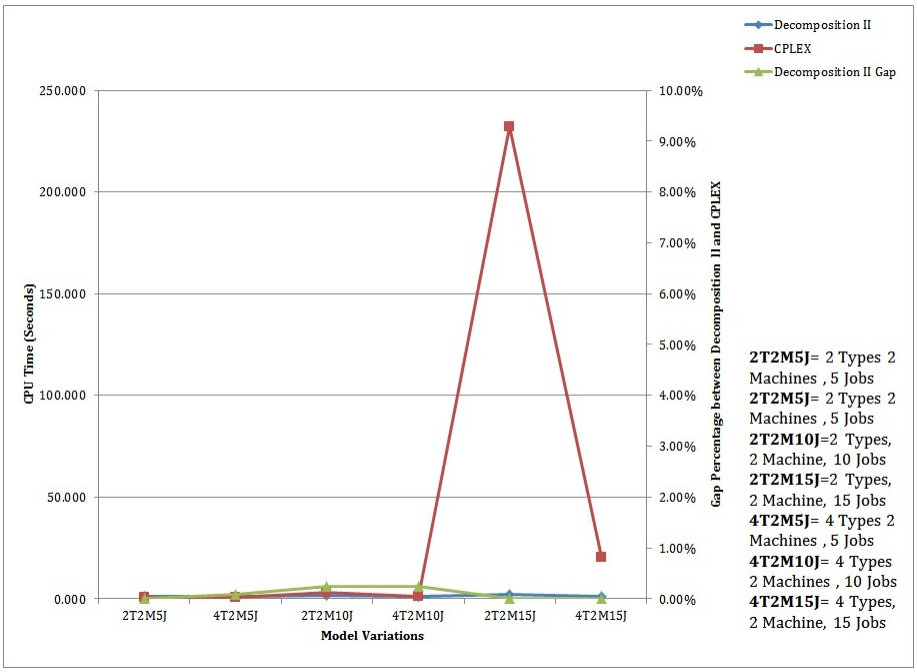


Figure 6.12: Capacitated Single Machine Stage 1 Agile Manufacturing: Average performance values for solutions obtained by Decomposition II and directly using CPLEX

Chapter 7

Concluding Remarks and Further Research

In order to meet the threats of tomorrow, the United States Army must be able to adjust its support functions as necessary. With the transition to a cyclic deployment schedule, referred to as ARFORGEN, the challenges for the generating force to support the operational force are great. In our research, we focus on improving the efficiency of the Initial Military Training such that the new soldiers are scheduled to graduate in line with ARFORGEN due windows for Brigade Combat Team requirements. Using lot streaming, we present a mixed-integer program of the Army Force Generation Problem (AFGP). This formulation was limited to the solution of small-sized problems obtained using CPLEX 12.2. The lot streaming problem that we consider is, to the best of our knowledge, the first attempt for a 3-stage lot streaming problem in an assembly configuration. From the literature review presented in Chapter 2, only a small portion of lot streaming research has focused on assembly configuration.

To improve upon our optimum seeking approach, we investigate the use of Dantzig-Wolfe decomposition. Specifically, we have observed that when the number of skill sets increases for a fixed total number of soldiers, the CPU time required decreases. The second stage (AIT) has separate resources per skill set and thus behaves as individual problems for each skill set. We have exploited this structure in developing a decomposition-based method, designated Decomposition II method, in Chapter 4. This method produces optimal solutions

in the presence of equal ready times and an un-capacitated first stage. The Decomposition II method applied to the AFGP, is therefore, a heuristic since the condition of equal ready times is violated in the AFGP case. From the results of computational investigations reported in Chapter 5, we observe the ability of Decomposition II to handle medium to large-sized problem instances in a reasonable amount of time. For small to medium-sized problem instances, we determined the accuracy of Decomposition II by comparing the solutions obtained with those attained by direct application of CPLEX as a benchmark. From our tests, the larger the problem instance, the more accurate the Decomposition II method is.

With a solution method that can handle large-sized problems, we apply Decomposition II to solve the realistic scenario of AFGP. The realistic AFGP scenario that we have considered uses historical data from FY09-FY10. With this data, we determine the most appropriate planning horizon that is based on the length of BCT due windows. Using the BCT windows as targets, we calculate the months of new recruits we must consider to fulfill the unit requirements within their due window. Given the number of recruits of the same skill set entering the system within the same month, we aggregate soldiers based on these two criteria. The result of aggregation amounts to a manageable problem size that we can solve by the Decomposition II method in a reasonable CPU time. Using this approach, our solution to the AFGP results in schedules with an estimated 28% reduction in mean flow time for soldiers in IMT over what is currently experienced in practice. Soldiers graduate from IMT better aligned with BCTs as all unit requirements are fulfilled without any soldiers being tardy. Our approach uses much stricter windows than currently enforced and still it meets the target requirements and due windows.

By formulating a solution approach to the AFGP, we have considered its variations that mirror some combination of the three stages. Using the configurations of Stages 1 and 2, we present, in Chapter 6, a new form of a hybrid flow shop with a common machine at Stage 1 and parallel machines for each job type at Stage 2. We address scenarios with an un-capacitated machine at Stage 1 as well as multiple un-capacitated machines at

Stage 1. We solve the problem using the Decomposition II method and compare the results with those obtained by direct application of CPLEX. For large-sized problem instances, the Decomposition II method required on average less time than that required by solving the problem directly using CPLEX. The accuracy of the Decomposition II method for this model configuration stays fairly steady with an optimality gap of roughly 2%.

We also present formulations for the hybrid flow shop with a single machine in Stage 1 that is capacitated (restricted sublot sizes). The solution method that we present uses the Decomposition II method. We present test results of the performances of both solution approaches. Overall, the performance of Decomposition II is consistent across the model variations. The solution obtained by the Decomposition II method are almost optimal for each model variation and the average CPU time requirements are roughly 2 seconds, including large-sized problem instances. The Decomposition II method can, therefore, handle large-sized problem instances more easily than direct solution by CPLEX and it gives fairly accurate solutions.

The first and third stages of the AFGP form a model configuration similar to the agile manufacturing environment. Stage 1 processes jobs of different types in batches that we refer to as sublots. Jobs move to Stage 2 to assemble into products requiring a set number of particular job types. We present formulations for three different model configurations of this agile manufacturing environment in Chapter 6. The first model configuration is exactly like Stages 1 and 3 of the AFGP, except that we minimize total completion time of the jobs as they form individual products. There also is a processing time required to assemble products. Stage 1 is a single un-capacitated machine. The performance behavior of solutions obtained directly using CPLEX and by Decomposition II are consistent with previous results. The two approaches require on average about the same CPU time for small and even medium-sized problem instances. For large-sized problem instances, Decomposition II requires less CPU time on average. The gap of the Decomposition II solution stays under 1% of optimality.

The second model configuration for agile manufacturing we present in Chapter 6 uses multiple machines in Stage 1 that are each un-capacitated. With multiple machines in Stage 1, the problem is more complex. While the Decomposition II method can handle problem instances of small, medium, and large sizes, the solutions obtained directly using CPLEX can only handle small and medium-size problem instances.

The third agile manufacturing model configuration we present in Chapter 6 has a single machine for Stage 1 with restrictions on the size of sublots that can be formed. Consistent with previous results, the Decomposition II method requires on average less CPU time than that required by direct application of CPLEX for large-sized problem instances. For each model configuration that we present in the agile manufacturing extensions of Chapter 6, the Decomposition II method out-performs the solution obtained by direct application of CPLEX for large-sized problems while producing almost optimal results.

Through our approach of aligning the institutional Army to effectively support ARFORGEN, we were able to reduce the mean flow time of Initial Military Training (IMT) while meeting BCT requirements. While our approach is a significant improvement to the current process, the overall problem is very complex. We specifically focused on new recruit classes and what their schedules should be to meet BCT requirements. Using a similar methodology, we can extend our research to address the other skill levels that are taught in AIT.

As extensions of the AFGP, we considered two different combinations of the three AFGP stages. We investigated a new type of hybrid flow shop with multiple un-capacitated machines in the first stage. For the Agile Manufacturing extension, we investigated the combination of the first and third stages for single and multiple un-capacitated machines and a single capacitated machine. For further research, we recommend consideration of multiple capacitated machines in the first stage, which would further generalize the problem domain within these manufacturing environments.

Bibliography

[1] Army War College. *How The Army Runs*. Carlise, PA, 2007.

[2] Baker, K. Lot streaming in the two-machine flow shop with setup times. *Annals of Operations Research 57*, 1-4 (1995), 1–11.

[3] Baker, K., and Pyke, D. Solution procedures for the lot-streaming problem. *Decision Sciences 21*, 3 (1990), 475–491.

[4] Baker, K. R., and Jia, D. A comparative study of lot streaming procedures. *Omega 21*, 5 (1993), 561–566.

[5] Bazaraa, M., Jarvis, J., and Sherali, H. *Linear programming and network flows*. John Wiley & Sons, Hoboken, NJ, 2010.

[6] Bedessem, N. Army Organizational Changes- The New Modular Army. In *An Army at War: Change in the Midst of Conflict* (Fort Leavenworth, Kansas, August 2005), Combat Studies Institute Press.

[7] Biskup, D., and Feldmann, M. Lot streaming with variable sublots: an integer programming formulation. *Journal of Operations Research Society 57* (2006), 296–303.

[8] Brah, S., and Hunsucker, J. Branch and bound algorithm for the flow shop with multiple processors. *European Journal of Operational Research 51*, 1 (1991), 88–99.

[9] Bukchin, J., Tzur, M., and Jaffe, M. Lot splitting to minimize average flow-time in a two-machine flow-shop. *IIE Transactions 34*, 11 (2002), 953.

[10] Bukchin, Y., Masin, M., and Kirshner, R. Modeling and analysis of multiobjective lot splitting for n-product m-machine flowshop lines. *Naval Research Logistics 57*, 4 (2010), 354–366.

[11] Bulbul, K., Kaminsky, P., and Yano, C. Flow shop scheduling with earliness, tardiness, and intermediate inventory holding costs. *Naval Research Logistics (NRL) 51*, 3 (2004), 407–445.

[12] Buscher, U., and Shen, L. An integrated tabu search algorithm for the lot streaming problem in job shops. *European Journal of Operational Research 199*, 2 (2009), 385–399.

[13] Buscher, U., and Shen, L. An Integer Programming Formulation for the Lot Streaming Problem in a Job Shop Environment with Setups. In *Proceedings of the International MultiConference of Engineers and Computer Scientists Vol II* (Hong Kong, March 2011), IMECS.

[14] Campbell, C. C. ARFORGEN: Maturing the Model, Refining the Process. *Army* (June 2009), 49–54.

[15] Cetinkaya, F., and Duman, M. Lot streaming in a two-machine mixed shop. *The International Journal of Advanced Manufacturing Technology* (2009).

[16] Chan, F., Wong, T., and Chan, L. Lot streaming for product assembly in job shop environment. *Robotics and Computer-Integrated Manufacturing 24*, 3 (2008), 321–331.

[17] Chen, J., and Steiner, G. Lot streaming with attached setups in three-machine flow shops. *IIE Transactions 30*, 3 (1998), 1075–1084.

[18] Crumbo, C. With army recruiting up, war waits for some. *Daily Press Nation and World* (26 July 2009), 14.

[19] Dastidar, S. G., and Nagi, R. Batch splitting in an assembly scheduling environment. *International Journal of Production Economics 105*, 2 (2007), 372–384. Scheduling in batch-processing industries and supply chains.

[20] Dauzere-Peres, S., and Lasserre, J.-B. Lot streaming in job-shop scheduling. *Operations Research 45*, 4 (1997), 584–595.

[21] Defersha, F. M., and Chen, M. A hybrid genetic algorithm for flowshop lot streaming with setups and variable sublots. *International Journal of Production Research 48*, 6 (2010), 1705–1726.

[22] Department of Army. *Army Posture Statement*. Washington, DC.

[23] Department of Army. *Army Posture Statement*. Washington, DC.

[24] Department of Army. *TRADOC Regulation 350-6: Initial Entry Training*.

[25] Department of Army. *TRADOC Regulation 350-10: Institutional Leader Training and Education*, 2002.

[26] Department of Army. *Addendum E: Army Force Generation Process*, 2008.

[27] Department of Defense. *Quadrennial Defense Review*.

[28] Department of Defense. *Time Phased Force and Deployment List*.

[29] Desrosiers, J., and Lubbecke, M. E. A primer in column generation. In *Column Generation*, G. Desaulniers, J. Desrosiers, and M. M. Solomon, Eds. Springer US, 2005, pp. 1–32.

[30] DeVor, R., Graves, R., and Mills, J. J. Agile manufacturing research: accomplishments and opportunities. *IIE Transactions 29* (1997), 813–823. 10.1023/A:1018575613893.

[31] Edis, R. S., and Ornek, A. Simulation analysis of lot streaming in job shops with transportation queue disciplines. *Simulation Modelling Practice and Theory 17*, 2 (2009), 442–453.

[32] Feldmann, M., and Biskup, D. Lot streaming in a multiple product permutation flow shop with intermingling. *International Journal of Production Research 46*, 1, 197–216.

[33] Glass, C. A., Gupta, J. N. D., and Potts, C. N. Lot streaming in three-stage production processes. *European Journal of Operational Research 75*, 2 (1994), 378–394.

[34] Glass, C. A., and Potts, C. N. Structural properties of lot streaming in a flow shop. *Mathematics of Operations Research 23*, 3 (1998), 624–639.

[35] Gunasekaran, A. Agile manufacturing: A framework for research and development. *International Journal of Production Economics 62*, 1-2 (1999), 87–105.

[36] Gupta, J. N., and Tunc, E. A. Schedules for a two-stage hybrid flowshop with parallel machines at the second stage. *International Journal of Production Research 29*, 7 (1991), 1489.

[37] Gupta, J. N., and Tunc, E. A. Scheduling a two-stage hybrid flowshop with separable setup and removal times. *European Journal of Operational Research 77*, 3 (1994), 415 – 428.

[38] Hall, N. G., Laporte, G., Selvarajah, E., and Sriskandarajah, C. Scheduling and lot streaming in two-machine open shops with no-wait in process. *Naval Research Logistics 52*, 3 (2005), 261–275.

[39] Huang, R.-H. Multi-objective job-shop scheduling with lot-splitting production. *International Journal of Production Economics 124*, 1 (2010), 206–213.

[40] Huisman, D., Jans, R., Peeters, M., and Wagelmans, A. P. Combining column generation and lagrangian relaxation. In *Column Generation*, G. Desaulniers, J. Desrosiers, and M. M. Solomon, Eds. Springer US, 2005, pp. 247–270.

[41] Johnson, M. The myth of the unsustainable army: An analysis of army deployments, the all volunteer force, and the army force generation model. Master's thesis, United States Army Command and General Staff College, 2009.

[42] Johnson, M. W. Army force generation within joint force provider, September 2005.

[43] Jungwattanakit, J., Reodecha, M., Chaovalitwongse, P., and Werner, F. Algorithms for flexible flow shop problems with unrelated parallel machines, setup times, and dual criteria. *International Journal of Advanced Manufacturing Technology 37*, 3/4 (2008), 354–370.

[44] Kalir, A., and Sarin, S. Optimal solutions for the single batch, flow shop, lot-streaming problem with equal sublots. *Decision Sciences 32*, 2 (2001), 387–398.

[45] Kalir, A. A., and Sarin, S. Constructing near optimal schedules for the flow shop lot streaming problem with sublot-attached setups. *Journal of Combinatorial Optimization 7* (2003), 23–44.

[46] Kalir, A. A., and Sarin, S. C. Evaluation of the potential benefits of lot streaming in flow-shop systems. *International Journal of Production Economics 66*, 2 (2000), 131–142.

[47] Kalir, A. A., and Sarin, S. C. A near-optimal heuristic for the sequencing problem in multiple-batch flow-shops with small equal sublots. *Omega 29*, 6 (2001), 577–584.

[48] Kim, J.-S., Kang, S.-H., and Lee, S. M. Transfer batch scheduling for a two-stage flowshop with identical parallel machines at each stage. *Omega 25*, 5 (1997), 547–555.

[49] Kropp, D. H., and Smunt, T. L. Optimal and heuristic models for lot splitting in a flow shop. *Decision Sciences 21*, 4 (1990), 691–709.

[50] Lee, G. Designs of components and manufacturing systems for agile manufacturing. *International Journal Production Research 36*, 4 (1998), 1023–1044.

[51] Lin, B. M. T., and Jeng, A. A. K. Parallel-machine batch scheduling to minimize the maximum lateness and the number of tardy jobs. *International Journal of Production Economics 91*, 2 (2004), 121–134.

[52] Liu, C.-H. Scheduling jobs with values exponentially deteriorating over time in a job shop environment. In *Proceedings of the International MultiConference of Engineers and Computer Scientists Vol II* (Hong Kong, March 2011), IMECS.

[53] Liu, J. Single-job lot streaming in m-1 two-stage hybrid flowshops. *European Journal of Operational Research 187*, 3 (2008), 1171–1183.

[54] Liu, S. A heuristic method for discrete lot streaming with variable sublots in a flow shop. *The International Journal of Advanced Manufacturing Technology 22*, 9-10 (2003), 662–668.

[55] Marimuthu, S., Ponnambalam, S., and Jawahar, N. Evolutionary algorithms for scheduling m-machine flow shop with lot streaming. *Robotics and Computer-Integrated Manufacturing 24*, 1 (2008), 125–139.

[56] Martin, C. H. A hybrid genetic algorithm/mathematical programming approach to the multi-family flowshop scheduling problem with lot streaming. *Omega 37*, 1 (2009), 126–137.

[57] McNeil, D. Army transformation & the army campaign plan overview, August 2006.

[58] Pinedo, M. *Scheduling Theory, Algorithms, and Systems*. Springer, New York,NY, 2008.

[59] Potts, C. N., and Baker, K. R. Flow shop scheduling with lot streaming. *Operations Research Letters 8*, 6 (1989), 297–303.

[60] Quinn, R. D., Causey, G. C., Merat, F. L., Sargent, D. M., Barendt, N. A., Newman, W. S., Velasco JR, V. B., Podgurski, A., Jo, J.-Y., Sterling, L. S., and Kim, Y. An agile manufacturing workcell design. *IIE Transactions 29* (1997), 901–909. 10.1023/A:1018587916619.

[61] Rahendran, C., and Chaudhuri, D. A multi-stage parallel-processor flowshop problem with minimum flowtime. *European Journal of Operational Research 57*, 1 (1992), 111–122.

[62] Ramasesh, R. V., Fu, H., Fong, D. K. H., and Hayya, J. C. Lot streaming in multistage production systems. *International Journal of Production Economics 66*, 3 (2000), 199–211.

[63] Sahni, S., and Gonzalez, T. P-complete approximation problems. *J. ACM 23* (July 1976), 555–565.

[64] Sanchez, L. M., and Nagi, R. A review of agile manufacturing systems. *International Journal Production Research 39*, 16 (2001), 3561–3600.

[65] Sarin, S. C., and Jaiprakash, P. *Flow Shop Lot Streaming*. Springer, New York, NY, 2007.

[66] Sarin, S. C., Kalir, A. A., and Chen, M. A single-lot, unified cost-based flow shop lot-streaming problem. *International Journal of Production Economics 113*, 1 (2008), 413–424.

[67] Training and Doctrine Command. *About TRADOC*, 2009.

[68] Trietsch, D., and Baker, K. R. Basic techniques for lot streaming. *Operations Research 41* (1993), 1065–1076.

[69] van den Akker, J., Hurkens, C., and Savelsbergh, M. Time-indexed formulations for machine scheduling problems: Column generation. *INFORMS Journal on Computing 12*, 2 (2000), 111–124.

[70] van den Akker, M., Hoogeveen, H., and Velde, S. Applying column generation to machine scheduling. In *Column Generation*, G. Desaulniers, J. Desrosiers, and M. M. Solomon, Eds. Springer US, 2005, pp. 303–330.

[71] Vanderbeck, F., and Savelsbergh, M. W. A generic view of dantzig-wolfe decomposition in mixed integer programming. *Operations Research Letters 34*, 3 (2006), 296–306.

[72] Vickson, R. G. Optimal lot streaming for multiple products in a two-machine flow shop. *European Journal of Operational Research 85*, 3 (1995), 556–575.

[73] Wallace, W. S. Victory starts here! changing TRADOC to meet the needs of the Army. *The U.S. Army Professional Writing Collection May-June* (2006).

[74] Wang, H. Flexible flow shop scheduling: optimum, heuristics and artificial intelligence solutions. *Expert Systems 22*, 2 (2005), 78–85.

[75] Wong, T., Chan, F. T., and Chan, L. A resource-constrained assembly job shop scheduling problem with lot streaming technique. *Computers & Industrial Engineering 57*, 3 (2009), 983–995.

[76] Zhang, W., Yin, C., Liu, J., and Linn, R. J. Multi-job lot streaming to minimize the mean completion time in m-1 hybrid flowshops. *International Journal of Production Economics 96*, 2 (2005), 189–200.

Appendix A: Acronym

Acronym	Definition
AFGP	Army Force Generation Problem
AIT	Advanced Individual Training
AOC	Area of Concentration
ARFORGEN	Army Force Generation
ASI	Additional Skill Identifier
AST	ARFORGEN Synchronization Tool
BCT	Brigade Combat Team
BOLC B	Basic Officer Leadership Course B
BT	Basic Combat Training
DOD	Department of Defense
FORSCOM	Forces Command
GWOT	Global War on Terror
HBCT	Heavy Brigade Combat Team
HRC	Human Resources Command
IMCOM	Installation Management Command
IMT	Initial Military Training
METL	Mission-Essential Task List
MFT	Mean Flow Time
MOS	Military Occupational Specialty
MTOE	Modified Table of Organization & Equipment
OCS	Officer Candidate School
OPLAN	Operations Plan
OSUT	One-Station Unit Training
QDR	Quadrennial Defense Review
ROTC	Reserve Officer Training Corps
TOE	Table of Organization & Equipment
TRADOC	Training & Doctrine Command
USAAC	United States Army Accessions Command
USMA	United States Military Academy