In Vivo Research Scheduling and Coordination in the Pharmaceutical Industry

By

Brent A. Hill

Bachelors of Science in Mechanical Engineering, Oregon State University, 1999

Submitted to the MIT Sloan School of Management and the Department of Mechanical Engineering in Partial Fulfillment of the Requirements for the Degrees of

Master of Business Administration

AND

Master of Science in Mechanical Engineering

In conjunction with the Leaders for Manufacturing Program at the

Massachusetts Institute of Technology

June 2008

Signature of Author ___

Department of Mechanical Engineering &
MIT Sloan School of Management
May 9th, 2008

Certified by ___

Stanley B. Gershwin, Thesis Supervisor
Senior Research Scientist, Department of Mechanical Engineering

Certified by ___

Andreas S. Schulz, Thesis Supervisor
Associate Professor of Operations Research, MIT Sloan School of Management

Accepted by ___

Professor Lallit Anand, Department of Mechanical Engineering
Chairman of Committee on Graduate Students

Accepted by ___

Debbie Berechman
Executive Director of MBA Program, MIT Sloan School of Management

This page has been intentionally left blank

In Vivo Research Scheduling and Coordination in the Pharmaceutical Industry

By

Brent A. Hill

Submitted to the MIT Sloan School of Management and the

Department of Mechanical Engineering on May 9th, 2008 in Partial Fulfillment of the

Requirements for the Degrees of Master of Business Administration and

Master of Science in Mechanical Engineering

ABSTRACT

The pharmaceutical industry is experiencing significant competitive pressures. Innovation productivity continues to decline, while the costs for drug R&D steadily rise. This project, sponsored by Novartis Institutes for BioMedical Research (NIBR), is intended to lower drug R&D costs and increase R&D process efficiency through improved research operations. This analysis focuses on improving the scheduling and coordination of early stage, in vivo drug discovery research projects within NIBR's animal facilities. Many of the communication processes used to coordinate research activities in these facilities use ad hoc methods for relaying critical information between research teams and the operations staff. Greater efficiencies can be achieved with the application of risk pooling concepts where dispersed research activities are brought together under a consolidated management structure. These efficiencies cannot be realized until the communication processes are improved. Integral to this improvement effort is the development of a fair and robust method for allocating in vivo resources to research projects using a centralized scheduling system.

This thesis provides the framework for developing a centralized scheduling system. The architecture of this tool requires a web-based interface in order to provide seamless access to the research community. Based on research workflows, the proposed tool coordinates input from scientists and uses this information to schedule the required resources. The complex constraints found in a research animal facility dictate the need for a unique scheduling approach. Adapted from existing airline gate scheduling research, this problem is formulated as a mixed integer linear program. A multi-criteria objective function uses the researcher's preference to optimize both room assignments and procedure start time. A Tabu search meta-heuristic has been developed to generate a near-optimal solution. The solution approach uses four neighborhood move strategies based on insert and interval exchange algorithms to optimize procedural room assignments. Although a functioning model was not developed, a recommended implementation plan is discussed.

Stanley B. Gershwin, Thesis Supervisor
Senior Research Scientist, Department of Mechanical Engineering

Andreas S. Schulz, Thesis Supervisor
Associate Professor of Operations Research, MIT Sloan School of Management

This page has been intentionally left blank

Acknowledgments

I would like to thank the Novartis Institutes for BioMedical Research for their overall support of the Leaders for Manufacturing program and for their support of my project specifically. While at NIBR, I interacted with many individuals who are very passionate about the development of life saving drugs, devoting a vast amount of their time and energy to this pursuit. I also found that many scientists and other individuals within the research community, especially the Laboratory Animal Services group, care deeply about the welfare of research animals and strive to provide the best possible care for these animals while they are involved in the drug discovery process. These individuals understand the sacrifices being made to better humanity and I respect them immensely for the work they perform. There are a number of people that I would like to recognize specifically for their contributions to this project:

Laboratory Animal Services and Animal Welfare:

- *Keith Astrofsky- You are truly passionate about running a first rate research animal facility. Thank you for spending countless hours discussing operational challenges and potential solutions to these challenges. I'm confident that you'll achieve success by staying positive and focusing on your goals.*
- *Peter Morgan- Always a source of expert information... thank you for all of the help.*
- *Nicole La Mattina-Smith- Your late addition to the team was critical for getting the project off the ground. Thank you for driving this project into the implementation phase.*
- *LAS Supervisors and Associates- Thank you for letting me work with you and teaching me about the interworkings of a vivarium.*
- *Colleen Cody and David DeOrnellis– I appreciate your help educating me on the various regulatory bodies that provide oversight for in vivo research, as well as NIBR's internal compliance processes.*

Research Operations and Facilities Planning: *I would like to thank Phil Plottel, Mike Golub and Alison Quigley for their dedication to the facilities analysis project. I'm confident that the solutions you helped to create will have positive operational impacts beyond this specific project.*

Disease Areas: *There are numerous individuals who devoted a lot of time and effort helping to teach me their processes and develop solutions to the challenges they face. Thank you for your support and I encourage you to embrace the processes discussed in this paper.*

Strategic External Resourcing:

- *Dan Warner- It was great working for you. Thank you for introducing me to the mountain biking trails around Boston.*
- *Dave Roberts- Thank you for helping establish the relationship with LFM and for recruiting me for this project. Your feedback and guidance on this project was a tremendous help.*
- *Brian Cain- Your humor is wicked-awesome. Just like G-mom's lemon bars!*

NIBR Executive Staff: *Jeff Elton- Thank you for sponsoring this project and making yourself available for project reviews. Having your support helped to make this a very successful internship for me.*

In addition, I would like to thank the Leaders for Manufacturing program at MIT for coordinating this internship process and providing me with a valuable learning experience:

- *I would like to thank my project advisors Andreas Schulz and Stan Gershwin for mentoring me during the thesis writing process. I appreciate you letting me leverage your expertise in the field of scheduling and optimization.*
- *I would like to recognize the LFM facility and staff for their dedication to the students in this program. Your involvement in LFM has helped to make this a great program.*
- *It has been a great experience getting to know my fellow LFM students over the last two years. The insights that everyone has provided have been a critical aspect of my learning experience. I wish you all the best of luck as we transition to the next phase of our lives.*

Finally, I would like to thank my parents Arnie and Linda Hill, as well as my sister Carrie Lane. Throughout my life, you have supported me during all of my endeavors (planned and unplanned), especially during my transition back into school. Specific to this project, your help with thesis proofreading made this a better document. You are the best family anyone could ever have and I'll never be able to thank you enough. I love you!

Table of Contents

Table of Figures

This page has been intentionally left blank

List of Tables

This page has been intentionally left blank

1. Introduction and Background

1.1. The Pharmaceutical Industry

The modern pharmaceutical industry has a diverse history that extends several centuries. Many large pharmaceutical companies of today were formally established in the 19th and 20th centuries. The discovery of drugs such as insulin (1921) and penicillin (1928) helped fuel the growth of this industry through World War II. In the wake of WWII, the pharmaceutical industry began investing heavily into research and development (R&D). These R&D efforts resulted in the development and mass marketing of many classes of new drugs including oral contraceptives, heart medications and blood pressure drugs. The drug Valium, discovered in 1960 and used to treat psychiatric conditions, quickly became the most widely prescribed drug in history when it was approved for commercial use in 1963. Many drugs, including Valium, began to be linked with negative side effects including dependency, habituation and birth defects. To address these concerns, agencies such as the US Food and Drug Administration (FDA) where created to oversee the drug development and approval process. The leading pharmaceutical companies, as measured by 2006 pharmaceutical sales, are shown in Figure 1.

LEADING PHARMACEUTICAL COMPANIES* — 2006

		SALES (BIL. $)				
	COMPANY	2003	2004	2005	2006	2007
1.	Pfizer	29.3	31.1	27.3	26.8	23.5
2.	GlaxoSmithKline	18.5	18.9	20.0	21.8	20.1
3.	Merck	14.0	15.3	15.4	16.7	17.6
4.	Johnson & Johnson	15.4	16.7	16.0	16.1	16.3
5.	AstraZeneca	10.1	11.5	12.7	14.7	15.5
6.	Amgen	7.7	9.7	11.9	14.5	14.3
7.	Novartis/Sandoz	10.5	11.6	13.0	13.9	13.9
8.	Hoffman-La Roche	5.3	6.2	8.2	10.4	12.3
9.	Sanofi-Aventis	9.0	10.2	11.1	11.0	10.9
10.	Lilly	7.7	8.2	8.7	9.2	10.3
	Total, top 10	127.5	139.4	144.3	155.1	154.7
	Total US market	219.6	239.9	253.9	276.1	286.5

*Pharmaceutical sales only.
Source: IMS Health Inc.

Figure 1- The Top 10 leading pharmaceutical companies[1]

In general, drug discovery and development is a high risk activity. On average, only 1 in 5000 compounds discovered ever reach consumers.[2] In addition, many products approved for

[1] Source: Standard & Poor's Industry Surveys, Healthcare: Pharmaceuticals, April 24th, 2008

[2] Ibid.

commercial distribution do not achieve commercial success. Less than a third of marketed drugs are able to recapture their R&D investment.[3] However, the introduction of a successful drug can be immensely profitable. As shown in Figure 2, the top ten drugs in the US grossed more than 41.8 billion USD in total sales for 2007.

TOP PRESCRIPTION DRUGS
(Ranked by 2007 US sales)

			SALES (BIL. $)				
DRUG	COMPANY	USE	2003	2004	2005	2006	2007
Lipitor	Pfizer	Cholesterol reducer	6.8	7.8	8.4	8.7	8.1
Nexium	AstraZeneca	Antiulcer	3.1	3.8	4.4	5.2	5.5
Advair Diskus	GlaxoSmithKline	Asthma	2.3	3.0	3.6	4.0	4.3
Plavix	Bristol-Myers Squibb/Sanofi	Antiplatelet	2.3	3.1	3.5	3.0	3.9
Seroquel	AstraZeneca	Antipsychotic	1.6	2.1	2.6	3.0	3.5
Singulair	Merck	Respiratory	1.8	2.2	2.5	3.0	3.4
Enbrel	Amgen	Antiarthritic	1.4	2.0	2.8	3.1	3.4
Prevacid	TAP	Antiulcer	4.1	3.9	3.8	3.6	3.4
Aranesp	Amgen	Antianemia	1.0	1.9	2.8	4.0	3.2
Epogen	Amgen	Antianemia	3.1	3.0	3.0	3.2	3.1

Source: IMS Health Inc.

Figure 2- Sales for top US prescription drugs[4]

The pharmaceutical industry is currently experiencing many competitive pressures that threaten this historically profitable industry. With prolonged development cycles, patent expiration has become an increasing problem since generic drug manufacturers typically capture significant market share when patents expire. In addition, large consolidated payers such as the US government (Medicare) and insurance companies are using their scale to pressure pharmaceutical companies for lower drug prices. This erodes their profit margins and decreases the drug manufacturer's ability to cover their R&D expenses. Perhaps the largest problem the pharmaceutical industry faces, though, is a decrease in innovation productivity amid rising R&D expenditures.

The period of time from 1970 through 1996 saw a general increase in the number of new molecular entities (NME) that were approved by the FDA. During this same period, spending on drug R&D also increased. Although R&D spending has continued to rise since 1996, the trend for NME approvals has reversed. The number of NMEs approved in US over the last decade has decreased significantly from 1996 levels, as shown in Figure 3. Globally, the number of drugs

[3] Source: Standard & Poor's Industry Surveys, Healthcare: Pharmaceuticals, April 24th, 2008

[4] Ibid.

approved with new active substances decreased more than twofold in the 1990s. During this same time, private-sector investment in drug R&D tripled.[5]

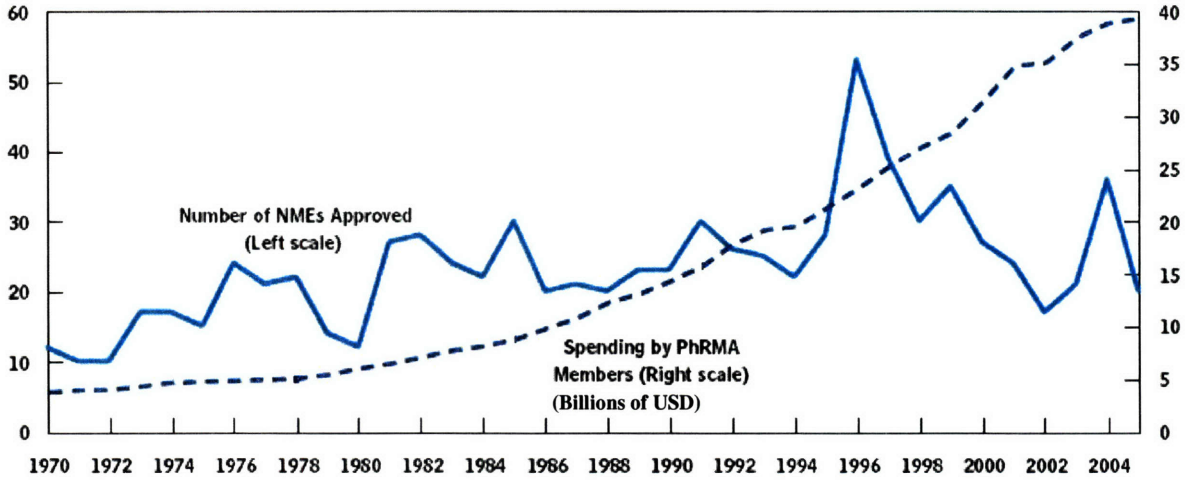


Figure 3- Drug innovation performance[6]

Entering 2008, the pharmaceutical industry continues on a downward trend. As of March 2008, the price-to-earnings ratios for large pharmaceutical companies were near a historic low of 13X (based on 2008 operating earnings per share). This represents a 15% discount compared to the boarder market, as measured by the S&P 500 index.[7] Most, if not all, of the major pharmaceutical companies are searching for solutions that will reverse these trends.

1.2. Novartis AG, NIBR and Drug Discovery

Novartis was created in 1996 through the merger of Ciba-Geigy and Sandoz, both Swiss based pharmaceutical companies. The corporate headquarters is located in Basel, Switzerland. At its core, Novartis is a pharmaceutical company and has divested several periphery business units in recent years to focus on four main areas. In addition to the pharmaceutical business unit that develops innovative drugs, Novartis has three additional business units. These include Vaccines and Diagnostics, Sandoz (generics) and the Consumer Health Group. The total workforce

---

[5] Source: European Federation of Pharmaceutical Industries and Associations, *The Pharmaceutical Industry in Figures- 2003 Update* (Brussels: EFPIA, 2003).

[6] Source: The Congress of the United States, Congressional Budget Office, *Research and Development in the Pharmaceutical Industry*, October 2006

[7] Source: Standard & Poor's Industry Surveys, Healthcare: Pharmaceuticals, April 24th, 2008

employed by Novartis is approximately 100,000 associates, which are located in more than 140 counties. The distribution of associates by both geography and business unit are shown below in Figure 4. As illustrated in Figure 2, Novartis is the 7th largest pharmaceutical company as measured by pharmaceutical sales. Led by Chairman and CEO Daniel Vasella, Novartis achieved 2007 net sales of 38,072 million USD and net income of 6,540 million USD, both based on continuing operations.[8]

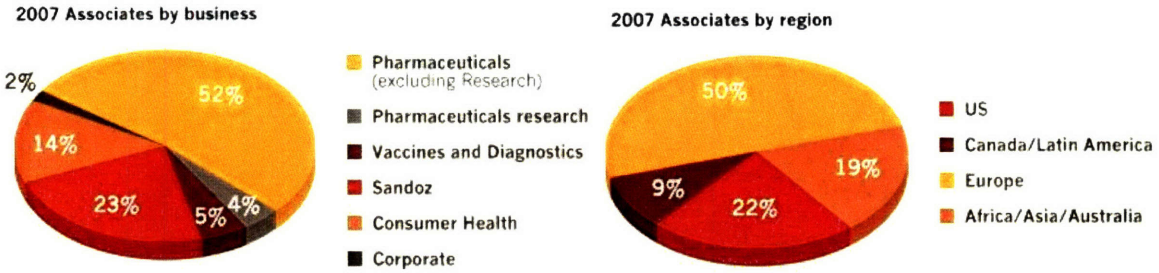


Figure 4- Novartis associate statistics[9]

In May 2002, Novartis announced the establishment of a new research center called the Novartis Institutes for BioMedical Research (NIBR). NIBR was established in order to reorganize and enhance Novartis' long tradition of drug discovery research and is now responsible for driving the drug discovery process. The global headquarters was established in Cambridge, MA in order to take advantage of talent through local universities, world class hospitals and to gain a presence in the world's largest pharmaceutical market. Novartis and NIBR have adopted a development process that is common in the industry, which is represented by the activities shown in Figure 5. Typically, NIBR will own the activities required to progress a drug development project from *Target Identification and Validation* through *Early Clinical Safety and Efficacy*.

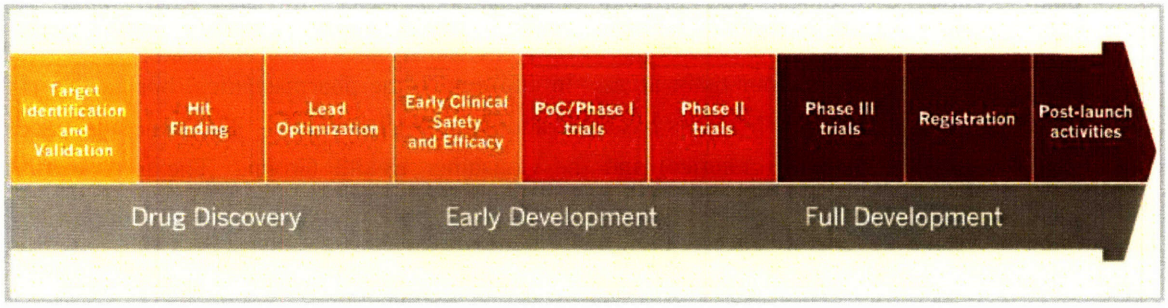


Figure 5- The drug discovery and development process[9]

---

[8] Source: Novartis 2007 Annual Report

[9] Source: Novartis website: http://www.novartis.com/

1.3. Vivarium Design and Operations

This paper focuses on in vivo research conducted at NIBR's Cambridge facility. In vivo refers to science that occurs within the living body of an animal. The drug discovery process has many milestone requirements that must be met in order to progress to the next stage in the drug development process. Prior to starting clinical trials, scientists must demonstrate that a low likelihood exists for a candidate drug to cause harm in human subjects. To ensure the safety of these human subjects, NIBR uses extensive animal testing to develop toxicological and safety pharmacological profiles for the compound of interest. Using results from animal testing to determine the likely response to a drug in the human body is called *Comparative Medicine.* Comparative medicine is a major area of research in both academia and privately funded pharmaceutical companies. Typically this research is undertaken during the *Lead Optimization* and *Early Clinical Safety and Efficacy* phases shown in Figure 5.

Special research animal facilities, called vivariums, house the animals that are used to conduct this research. Good animal management and human health protection require separation of animal facilities from personnel areas, including labs and offices.[10] Although the specific information regarding these facilities is confidential, NIBR operates two vivariums in the Cambridge area. Each facility is self contained and can accommodate the vast majority of in vivo research generated by NIBR's Cambridge based scientific community. While working at NIBR, I observed that the drug discovery process is very fickle. Scientists test many different compounds and are constantly altering the compounds to determine if enhanced properties can be obtained. Many times, scientists will pursue a compound that initially appears to be promising, but later is found to have adverse characteristics either in lab tests (in vitro) or in vivo. This trial and error approach requires the research process to be flexible and scalable in order to accommodate the changing research needs.

To accommodate these requirements, the drug discovery process typically uses small animals that can be housed in relatively high concentrations and are easy to maintain. Rodents, such as mice and rats, meet these criteria and are also well understood from a scientific standpoint. The vast majority of animals at NIBR's two vivariums are mice and rats. Other species that can be found in the vivariums include rabbits, cotton rats, ferrets, guinea pigs and hamsters. Robust and repeatable vivarium management processes are required in order to produce reliable and accurate research results. Central to these processes is the manner in which the animals are housed. If the semi-unique identities of these animals are not maintained, data collected throughout this process will be confounded. The term semi-unique is used since rodents are typically kept in cages that

---

[10] National Research Council. (2002). *Guide for the Care and Use of Laboratory Animals.* Washington D.C.: National Academy Press.

contain more than one animal. For mice, a maximum of four animals can be housed in one cage, while a maximum of two rats can be kept in a single cage. Although some male rodents can become aggressive when placed in a cage with another male, group housing of animals is typically beneficial since it provides companionship and interaction for the animals.

Many different cage and rack designs exist in the industry. NIBR primarily uses individually ventilated cage (IVC) technology, which provides a fully contained environment that prevents the introduction or escape of micro-organisms. These cages use High Efficiency-Particulate Air (HEPA) filters to provide a barrier between the sterilized inner cage and the external environment. IVC cages improve the quality of research by reducing the likelihood that animals will transmit pathogens between cages. As will be discussed in the next section, the introduction of pathogens into an in vivo research environment can devastate ongoing research, costing millions of dollars in lost animals and delayed research. A typical cage design is shown in Figure 6 and contains four primary components: lid, base, wire-bar food hopper and water bottle. The picture on the right illustrates the flow of air within the cage. The blue arrows represent HEPA filtered inlet air, while the exhaust air is depicted by the red arrows. The exhaust air can either be HEPA filtered and recirculated into the holding room or directly exhausted to the outside environment.

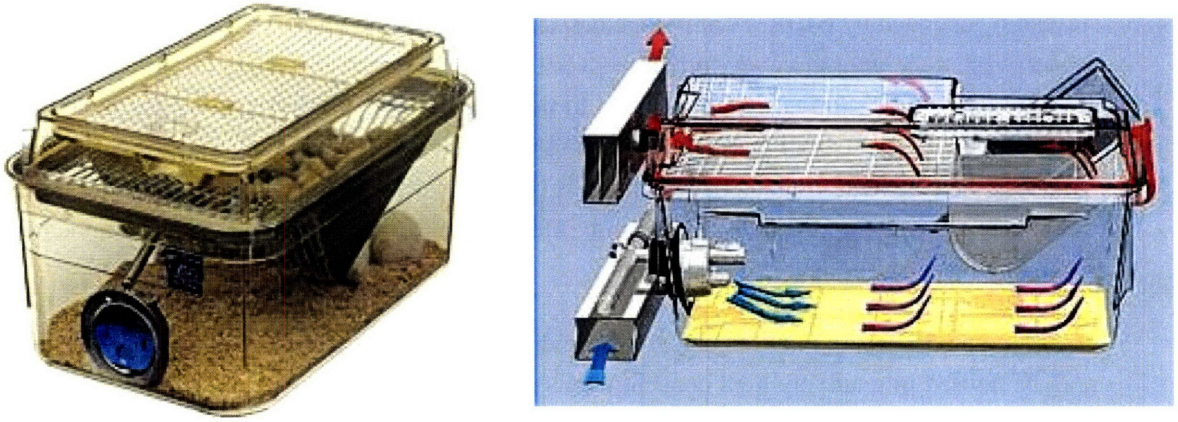


Figure 6- Typical rodent cage design and air flow simulation[11]

To achieve scale efficiencies, cages are placed in racks that can hold as many as 140 cages (double sided, mouse). Figure 7 illustrates common rack designs. Typically the cage will mate with two rails and slide into a closed (locked) position on the rack. As the cage mates with the rack, the mechanism that automatically provides water to the animals passes through a small opening in one side of the cage (directly above the air inlet). Racks are typically mounted on

[11] Source: Allentown, Inc. http://www.allentowninc.com/

casters, which allow researchers and vivarium managers to move the cages within a holding room. In general, racks are built from stainless steel since these components must withstand harsh cleaning agents and high temperatures. In addition, cages must be fabricated from high temperature plastic since they are autoclaved as part of the cleaning process, which occurs at least every 10 days at NIBR.

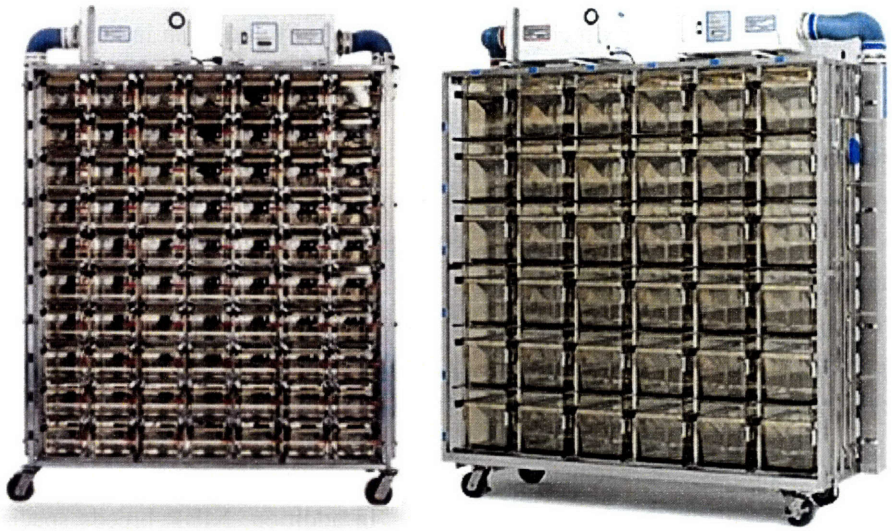


Figure 7- Typical mouse and rat racks with caging[12]

The area within a research animal facility has many uses. Space requirements can be divided into two primary categories: program space and non-program space.[13] Non-program space consists of circulation (corridors, elevators, etc), mechanical and architectural (wall dimensions, etc). Program space can be subdivided into five categories:

1. *Animal Holding:* Holding rooms and isolation areas
2. *Procedural:* Space where scientists conduct research activities
3. *Service:* Central animal husbandry[14] areas
4. *Personal:* Offices, locker rooms, meeting areas
5. *Staging:* Areas were small quantities of materials are stored for daily use

---

[12] Source: Allentown, Inc. http://www.allentowninc.com/

[13] Ruys, T. (1991). *Handbook of Facilities Planning, Volume 2: Laboratory Animal Facilities.* New York: Van Nostrand Reinhold

[14] Animal husbandry is the science of taking care of animals. Important husbandry activities include feeding animals, changing cages on a regular basis, and performing miscellaneous veterinary functions.

When designing a facility, planners should attempt to maximize the amount of space dedicated to animal holding and procedural activities, while ensuring that operational needs in the other three categories are met. This goal is in line with lean operating principles. Typically operations in a vivarium are centered on the service area (also called cage wash), which is divided into three areas: dirty, clean and sterile. When cages are changed in the holding rooms, the soiled material will be transported to the dirty cage wash area. The contents of a cage will be emptied into a disposal container, after which the cage is processed through an industrial washing system. The dirty cage wash is considered the most contaminated area in a research animal facility and great care is taken to ensure contaminants do not leave this area. For this reason, travel is not allowed between dirty cage wash and clean cage wash (space on the receiving side of the industrial wash system) areas. Once the cleaned cage is retrieved from the industrial wash system, bedding is added and the cage is placed in inventory. When additional clean cages are ordered by the husbandry staff, the cages in inventory are passed through an autoclave in order to ensure the contents are free of containments. These cages are placed in the sterile area until they are retrieved by the husbandry staff. Many facilities also use clean/dirty corridor systems to transport either fresh or soiled material between the holding rooms and central service area.

Research animals are obtained through two methods. Third party vendors breed animals offsite and fulfill demand on an as needed basis. In addition, NIBR breeds special mice strains that cannot be obtained from third party vendors. When animals are received from a third party vendor, the packaging is sterilized and the animals are transferred to a holding room. The vivarium management staff will determine which holding room to place the animals in based on the individual (or group) performing the research. Once inside the holding room, the animals are transferred into clean cages and placed on racks. Holding rooms vary in size and design. Very small rooms called *cubicles* can be used when housing small quantities of mixed species. Cubicles are small, module rooms that share a common circulation and service space. In contrast to cubicles, holding rooms with an area greater than 600ft^2 are used when research is being conducted on large quantities of animals of the same species. These large rooms are typically used for holding breeding colonies. Most holding rooms in the NIBR vivariums are medium size, with floor space ranging between 200-600ft^2. Two holding room layouts have been included to illustrate a typical room design. Figure 8 shows the layout of a holding room that uses standard racks with cages on two-sides (similar to the design shown in Figure 7). In addition, Figure 9 is based on a rotating mouse rack design. This design allows scientists and vivarium staff to access the cages from one side. As a result, the rack can be moved against the wall, which allows for better space utilization. Animal Care Systems (ACS), Inc. supplies racks and cages based on this space saving design. The ACS rack and cage are shown in Figure 10.

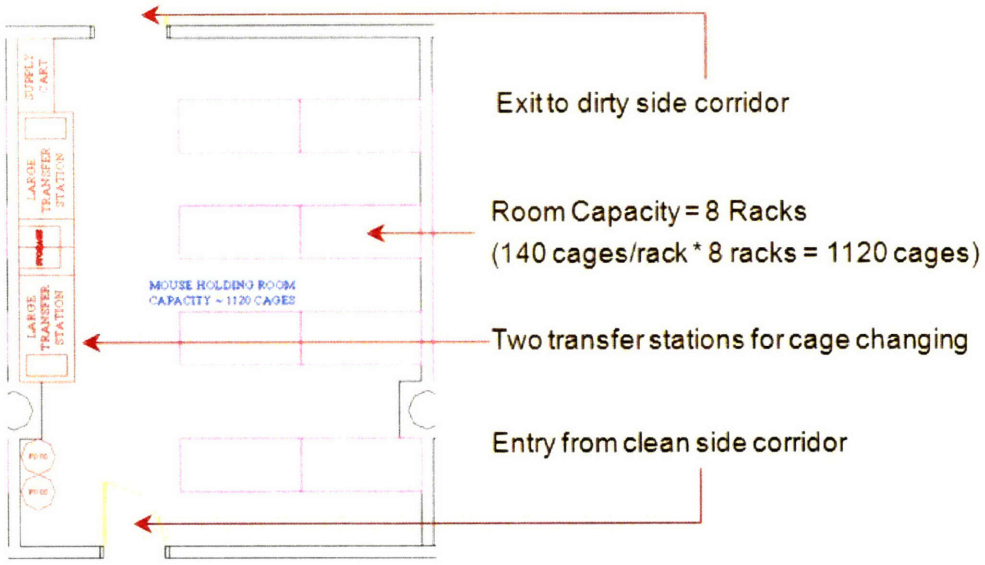


Figure 8- Typical mouse holding room with standard, two-sided racks

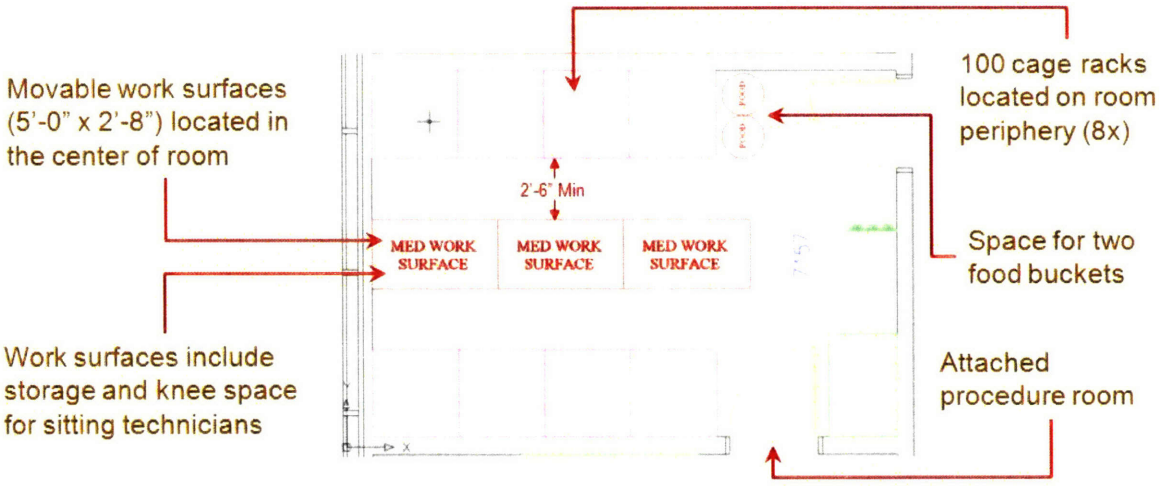


Figure 9- Typical mouse holding room with OptiMICE caging system

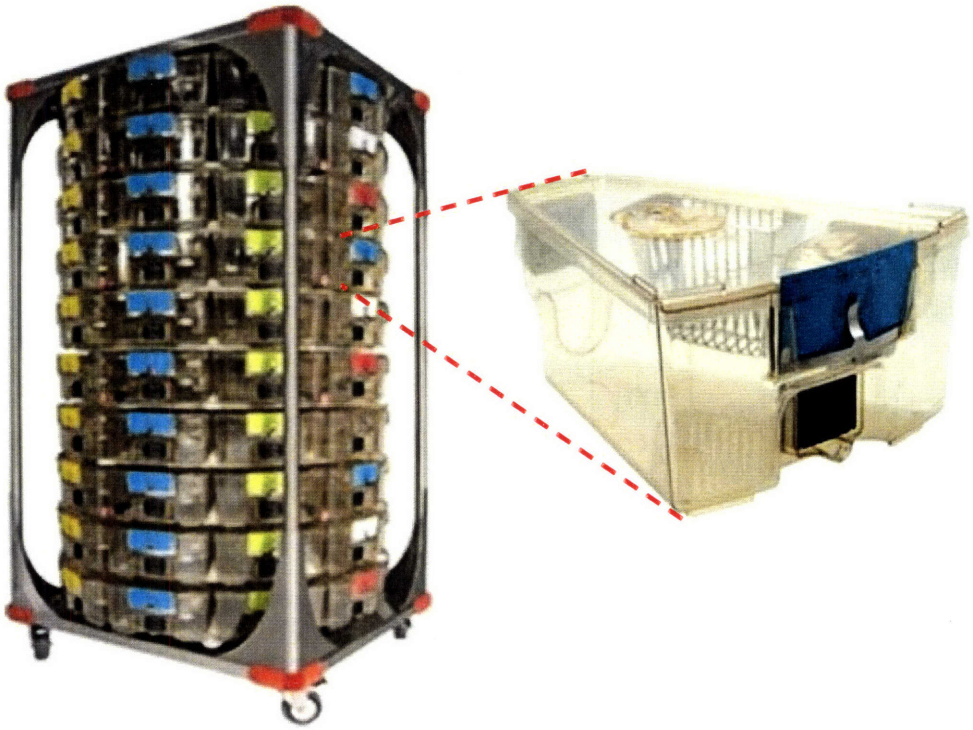
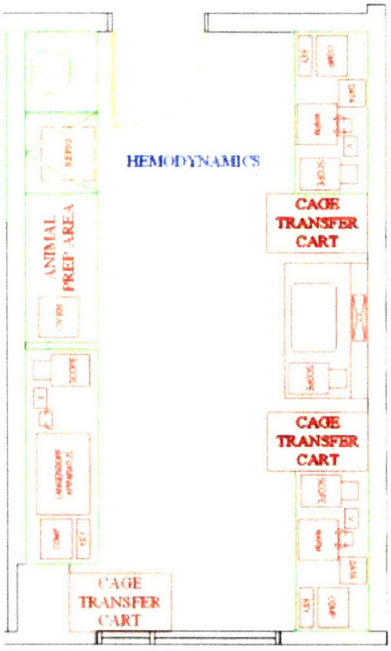


Figure 10- ACS OptiMICE caging system with rotating rack design[15]

Procedure rooms provide the infrastructure required to perform various research activities. Procedure rooms are very diverse in nature, differing in both size and layout. Some rooms are designed to accommodate general procedures that do not require unique equipment. These rooms typically have work surfaces on at least two walls and contain general purpose equipment such as scales, centrifuges and microscopes. Other procedure rooms are designed for a specific function. Many activities in a research animal facility require complex instrumentation to collect data. This equipment includes imaging devices (MRI, ultrasound, etc), lasers and metabolism measurement devices. Other rooms are designed to accommodate surgical and terminal procedures. The equipment for these rooms dictates the room layout and design. Figure 11 illustrates a typical hemodynamics room layout. Three workstations are accommodated in this layout, along with an area where animals can be prepared for the procedure. In this case, procedures are terminal, so a compact necropsy workstation has been included in order to enable tissue collection without removal of animal carcasses from the room. As this room design illustrates, multiple projects can be processed in a room at one time. For this reason, care must be taken to differentiate the procedure room from the procedure space. In this example, each workstation is a separate procedural space. The importance of this distinction will be highlighted later in the paper.

---

[15] Source: Animal Care Systems, Inc. http://www.animalcaresystems.com/



Hemodynamics Procedure Room Requirements:

- Two hemodynamics/surgical stations
- One Langendorf station
- One animal prep station that includes a vacuum system and 8' countertop space
- Necropsy table with backdraft ventilation
- Microscopes at all workbenches
- Bench top oven
- Refrigerator
- Sink
- Three cage transfer carts

Figure 11- Typical procedure room layout, including equipment for terminal procedures

In general, two groups work together to conduct research in a vivarium. First, the research scientists and their teams manage execution of research projects. This includes submitting animal orders, preparing for procedures and conducting actual procedures. Data collected during these activities is managed and owned by the research teams. Typically each project team will be led by a *primary investigator* (PI), who is a senior research scientist. The PI is ultimately responsible for delivering project results. At NIBR, the research teams are organized into *disease areas*, which focus on developing therapeutic drugs specific to these diseases. Examples of disease areas include *Oncology*, *Ophthalmology* and *Cardiovascular*. Each disease area is assigned procedural and holding rooms from which they coordinate their research.

In addition to the research scientists, the vivarium management team coordinates operations within the vivarium. At NIBR, this group is named *Laboratory Animal Services* (LAS). LAS is responsible for ensuring the proper care and wellbeing of research animals. This includes performing husbandry functions, periodically checking animals to ensure they are in good health and maintaining a functioning facility. LAS also coordinates procurement of animals from third party suppliers. The LAS team consists of a program director who also functions as the clinical veterinarian, a database/information manager, four supervisors and a large team of associates. The associates are divided into small teams that focus on husbandry, cage wash and facilities upkeep. LAS also performs some functions which extend into the research arena. These activities include breeding colony management and study support services. Study support involves LAS personnel performing research that has been requested by the disease areas. This can include working side-by-side with the research teams to collect data or owning the execution

of an entire procedure. For example, LAS is frequently asked to dose animals with study compounds and collect blood samples. LAS has highly skilled technicians who are capable of performing many procedures. Study support is important since it allows the LAS group to hone their skills and perform services in an efficient manner.

In general, research conducted in the animal research facilities is dictated by protocols that are developed by the research team under the supervision of the PI. These protocols identify the purpose of the research activities, which includes the rationale and objectives for conducting specific procedures. The protocol dictates the species and quantity of animals that will be used to conduct the research. These numbers are typically justified using statistical methods. The research team is required to identify specific individuals who will be conducting research under a protocol and the procedures that these individuals have been qualified to perform. Prior to a team beginning research in the animal facility, the protocol must be approved by the *Institutional Animal Care and Use Committee* (IACUC). The IACUC is a cross functional team that is comprised of representatives from the major research groups, the Animal Welfare team, senior NIBR management and external representatives. Additional information regarding protocols and the IACUC approval process can be found in *The Guide*.[16]

1.4. Challenges and Constraints

Coordinating research operations within an animal facility is a difficult task. A robust operational program will accommodate the needs of the research community, rules imposed by regulatory bodies and operational objectives. Operational objectives include schedule efficiency, research quality, and cost efficiency. A handful of government and private institutions monitor and regulate research operations within NIBR's vivariums. The most basic regulatory requirements are imposed by the US Department of Agriculture (USDA), which oversees regulations specified in *The Animal Welfare Act*.[17] *The Animal Welfare Act* does not address smaller animals including rodents such as mice and rats. To address this shortcoming, the National Research Council developed *The Guide*, which covers all species in a variety of research environments. *The Guide* provides a high level framework for conducting research in an animal facility and does not address a plethora of detailed questions concerning day-to-day operations. Regulations and recommended procedures for day-to-day operations are provided by the Association for Assessment and Accreditation of Laboratory Animal Care (AAALAC). AAALAC is a private, nonprofit organization that promotes the humane treatment of animals in science through

---

[16] National Research Council. (2002). *Guide for the Care and Use of Laboratory Animals.* Washington D.C.: National Academy Press.

[17] USDA regulations can be found in the *Animal Welfare Act and Animal Welfare Regulations* guidebook, November, 2005

voluntary accreditation and assessment programs. NIBR is a member of AAALAC and just recently passed an accreditation inspection. Local authorities, including the City of Cambridge, also provide oversight of animal research activities. In general, these groups have developed many regulations and guidelines that must be considered when creating and implementing an operations plan for an animal facility.

The scientists performing in vivo research demand high quality service. This is understandable since the data they are collecting will be used to determine if a component is safe for use in humans. Confounded data can potentially cause huge financial loses and more importantly, adverse effects on human subjects during clinical trials. In addition, my observations suggest that the scientific community has been conditioned to believe that their research should not be inhibited by operational concerns or competing research projects. This mentality has been reinforced through the allocation of research space. Many in the research community feel that they own the space within the vivarium that their group has been assigned. Limited interaction between research groups occurs due to this division of research facilities.

Allocation of resources in this manner is problematic since the research conducted at NIBR is highly variable. Since many resources are not shared across research teams, fluctuating demand will cause both localized resource shortages and a high occurrence of idle resource capacity. Discussions with scientists who work in various Disease Areas revealed that coordination of resources is inadequate and many of these scientists do not have access to resources that would both streamline and enhance their research. A similar challenge is found in supply chain systems, where fluctuating demand will create material shortages and high inventory levels. Material shortages impact customer service level, while high inventory levels require greater working capital to maintain. A concept called risk pooling has been developed to address the challenge of fluctuating demand. In supply chains, risk pooling involves combining inventories into a single warehouse from which customers are serviced. By aggregating the demand into a single facility, the overall standard deviation can be reduced, which in turn decreases the safety stock that needs to be held.

The same concepts can be applied to a research animal facility. By bringing together decentralized research projects, demand fluctuations can be smoothed and research teams can be given greater access to in vivo resources. Care must be taken to ensure all constraints and regulations imposed by the research community and oversight organizations are accommodated. This includes keeping different species in separate holding and procedural areas, supporting environmental requirements (light cycles, noise levels, etc) and ensuring animals are placed in areas that match the biohazard level of the research being conducted. The potential benefit of risk pooling in a research environment is illustrated in Figure 12. The graph contains demand over 52 weeks from four separate disease areas. The animals being compared are mice that are in a low biohazard level environment (i.e. BL1) and on normal light cycles. As this graph shows, DA1 (blue) and DA2 (yellow) have a relatively high level of demand fluctuation over the time interval.

This is contrasted with DA3 (green) and DA4 (purple), both of which have fairly mild fluctuations in their demand patterns. The red line indicates the aggregated demand for these four groups. In order to determine the total cage capacity required to conduct research, the "safety stock" needs to be added to the average research demand. Safety stock in this case is the total number of cages required above the average demand to successfully accommodate research activities. For this analysis, three standard deviations were used to determine the safety stock. When the research groups operate in a decentralized model, a total of 2340 cages are required to provide a sufficient level of service. However, when the demand is aggregated, the total number of cages decreases to 1949. This represents a reduction of 391 cages or 17%.

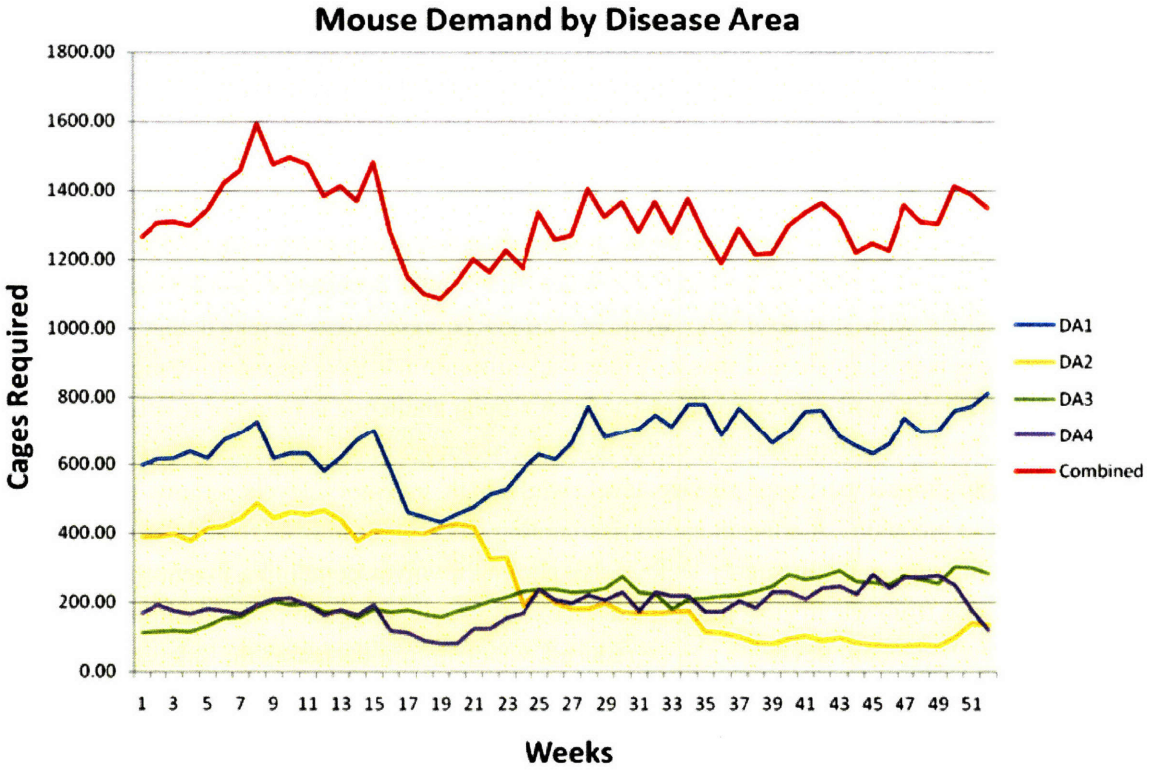


Figure 12- Annual mouse demand fluctuations by disease area

Behavior of the research community will also impact the ability to prevent infectious diseases from entering the research animal facilities. Introduction of infectious pathogens into an animal facility can significantly impact the research being conducted in these facilities. Keith Astrofsky, the lead clinical veterinarian at NIBR's Cambridge campus, discussed his assessment of biosecurity at NIBR's Cambridge vivariums in a 2006 report (Astrofsky, 2006). The intent of his report was to assess risks to the uninterrupted conduct of in vivo research operations. This report provided a comprehensive overview of current biosecurity threats, as well as the actions

recommended to address these threats. In his analysis, Astrofsky identified four areas of concern linked with behaviors of the research community:

1. Movement of animals between rooms and facilities
2. Movement of experimental-related equipment between rooms and facilities
3. Introduction of contaminated biologicals into research animals within the vivarium
4. Personnel trafficking

All four concerns are partially controllable through robust operational practices. The research community must follow best practices related to biosecurity. In addition, robust processes must be developed that support these best practices and create an environment where these best practices can easily be adopted. In his report, Astrofsky indicates that these processes require continued development and refinement.

1.5. Thesis Motivation and Scope of Research

As discussed in the opening section, the pharmaceutical industry is experiencing significant competitive pressures. In addition to increased competition from generic drug manufacturers and pricing demands from centralized payers, the industry has also experienced a significant decline in innovation productivity. This decline in innovation has occurred while costs for drug R&D have increased. Acting on these trends, NIBR's executive management believe that significant operational improvements can be made within their research animal facilities. Improving the operations will both lower drug R&D costs and improve the efficiency of the R&D process.

These changes cannot be superficial and must address the systemic issues driving operational inefficiencies. At the heart of these concerns are the methods used to allocate resources to the research community. The discussion regarding risk pooling highlights the opportunities that exist to better utilize NIBR resources, while simultaneously improving research output. Many of the communication processes in place today use ad hoc methods for relaying critical information between research teams and the vivarium staff. These communication methods are both time consuming and create significant opportunities for data corruption. In short, the research community has an in vivo R&D process that works, but not well. Developing robust processes will require additional infrastructure, as well as trust between the research community and LAS. In order to build this trust, LAS must have the processes to provide reliable and efficient vivarium management services. The remaining material in this thesis focuses on developing a fair and robust method for allocating in vivo resources to research projects using a centralized management system. By implementing this management system, LAS will have the ability to build a foundation on which it can begin developing the relationships required to drive process efficiencies. These efficiencies include both consolidation of research activities (i.e. risk pooling) and greater management of research activities, including increased colony management and study support.

A significant amount of effort has been spent understanding the complex research process that occurs within NIBR's animal facilities. Numerous constraints and research requirements dictate the need for a customized process to manage and coordinate research activities. The intent of this thesis is to outline these constraints and develop a framework that can be used to allocate resources in a reliable fashion. This framework has been designed in a manner that allows for integration into an internet based tool (i.e. web-based). The intent of this tool is to provide a system that is easy to use, provides system wide resource optimization and allows managers to better align research incentives with in vivo operational goals. Due to the complex nature of in vivo research, only constraints related to NIBR's Cambridge vivariums were considered. In addition, this research does not attempt to implement the scheduling and coordination framework into a validated web-based tool. The information contained in this paper should be used by NIBR to develop this tool and related processes.

1.6. Thesis Outline

This thesis is organized into the following chapters:

Chapter 2 discusses the nature of in vivo drug research and proposes a system architecture for coordinating and scheduling this research. The concept for a web-based scheduling tool is discussed in detail.

Chapter 3 provides an overview of scheduling methods that have successfully been used outside the pharmaceutical industry. These models are leveraged to produce a mixed integer linear programming model with a multi-criteria objective function. This model is built to allocate in vivo research resources while minimizing logistical and scheduling impacts.

Chapter 4 focuses on the development of a heuristic based algorithm that can be used to realistically manage the data requirements of this system. A tabu search algorithm is proposed, which leverages four move strategies and an iterative search technique. A series of subroutines are proposed which are used to execute this algorithm.

Chapter 5 summarizes the recommendations made in this thesis and highlights areas for additional research.

2. Scheduling Tool Development

2.1. Chapter Overview

Developing a method for scheduling and coordinating research must take into account scientist preferences, research structure and detailed research constraints. This chapter focuses on preferences and structure by developing a system architecture for coordinating and scheduling in vivo research. The concept for a web-based scheduling tool is discussed in detail. This discussion begins with an overview of the challenges associated with managing in vivo research and addresses critical features that should be included in a scheduling tool. The nature of typical research design is discussed and this structure is translated into workflows that can be managed with a centralized planning system. The chapter concludes with an overview of a conceptual tool design. The detailed function of this tool is discussed and several examples are used to relate high level concepts with tangible examples. Algorithms discussed later in the paper will leverage the process frameworks developed in this chapter.

2.2. Scheduling Tool Motivation and Challenges

Many research activities conducted within NIBR's Cambridge vivariums are scheduled using ad hoc processes. These processes range from email based calendaring to paper based schedules that hang in the actual rooms. In some cases, a scheduling process does not exist since individual researchers "own" a procedural room and are guaranteed access to this space. The decentralized nature of these processes makes system level coordination of space and resource allocation very difficult. Changing the operating model from the current user controlled system to a centralized operation has the potential to create more efficiency and costs savings. Unfortunately, researchers will not allow this transition to occur without assurance that resources will be allocated in a manner that is unbiased. In addition, scientists insist they retain the ability to specify both the preferred procedural space and the procedural start time. These demands require the development of a tool for coordinating data entry and organizing the data once it is obtained.

The research environment presents many operational challenges, which will need to be accounted for in a scheduling tool. First, vivarium resource demand from individual scientists or small research organizations typically has a high degree of variability. Having the ability to aggregate this demand (i.e. risk pooling) is one of the major efficiency drivers. Successfully implementing a risk pooling operational strategy has two key challenges. First, the research organizations will be resistive to sharing resources with other research groups. Although this thesis does not address this specific challenge in detail, my experience suggests that agreement can be reached by highlighting the positive benefits of centrally organized research operations. Many scientists complain that they don't have access to equipment and other resources (e.g. necropsy facilities). In some cases, scientists don't know what equipment already exists in the vivariums. A

centralized system that coordinates allocation of these resources provides superior transparency compared to the current process. Although scientist will have to prioritize planning activities, they will have access to a wider range of equipment and resources. In many cases, the opportunity to use these resources will outweigh the perceived hassle of additional planning. The second challenge involves understanding the complex constraints that exist in a typical in vivo research environment. As discussed in Section 1.3, there are three types of constraints that must be considered: researcher imposed, government imposed and facility driven. A scheduling and coordination tool that ineffectively addresses any of these three constraint categories will result in the inability to drive adoption of the scheduling process with the research community.

The dynamic nature of in vivo research also poses a challenge for coordinating research activities. A significant number of scientists plan their research based on the results of ongoing experiments. In some cases, data collected during the end of one week will drive activities for the following week. Although most scheduling systems benefits from having accurate data provided in a timely manner, the system implemented in this environment must be flexible enough to accommodate short lead-time requests and modifications to existing requests. In order to develop an optimized room assignment schedule, the scheduling algorithm will need to be executed at a specific point in time. The timing of the assignment process (i.e. when the algorithm determines room assignments) will depend on the average assignment lead-time. This lead-time exists since a portion of scientists and their teams need to prepare for upcoming procedures. In some cases, this preparation is room dependent. The incentives created by this lead-time will be very important for encouraging behaviors that benefits a centralized planning system. A scheduling "deadline" that is too far in advance of the procedural start time will result in scientists rejecting the entire process. Conversely, a deadline that is too close to the procedural day will result in complaints from scientists who do not have enough time to plan their procedures. As a general process rule, scientists who follow planning guidelines and submit their requests in the proper timeframe should not be negatively impacted by other scientists who do not prioritize the planning process and have difficulty submitting their requests in a timely manner.

2.3. Research Activities

Understanding the methods used to plan research activities is critical for developing a scheduling tool. In order to enable adoption of the tool, it is important to implement a data entry process that aligns with the methods used to plan these activities. In general, scientists view their research as a linear set of activities. These activities begin when research animals are ordered and are completed when the animals are euthanized. Within this continuum are discrete time points at which the animals begin various procedures. The procedures are separated by periods of time during which animals recover from procedures (e.g. surgery) or are conditioned for future research. Conditioning can include gestation of a disease or the development of physical attributes that are integral to the research of interest (e.g. an increase in body fat for cardiovascular research).

Although research is generally viewed as a linear set of activities, these activities can significantly differ in both quantity and frequency. For example, some research activities are designed to measure the pharmacokinetics or pharmacodynamics of particular drugs. These studies typically have only three activities (animal ordering, dosing/blood drawing and euthanizing) and are completed in a short amount of time (typically five to seven days). Other research activities can occur over months and even years. Take for example the research activities that are shown in Figure 13. This timeline outlines research for a compound intended to reduce atherosclerosis, which is a leading cause of cardiovascular disease. This particular research uses rabbit models to test the effectiveness of various compounds. As the timeline indicates, activities occur over a period of 22 weeks and begin when the animals are ordered. After the animals arrive in the facility, their baseline lipids (fatty substances, including cholesterol and triglycerides) are measured. The animals then undergo a surgical procedure called balloon angioplasty, which makes their arteries more susceptible to the development of atherosclerosis. Time is allotted for surgical recovery, during which the rabbits are transitioned to a high fat, high cholesterol diet. Once the animals have fully transitioned to this diet, they undergo intermittent lipid analysis and Magnetic Resonance Imaging (MRI) analysis to determine the effectiveness of the compound in reducing the occurrence of atherosclerosis. At completion of the study, the animals are euthanized and a full histology (the study of tissues and cells) is completed.

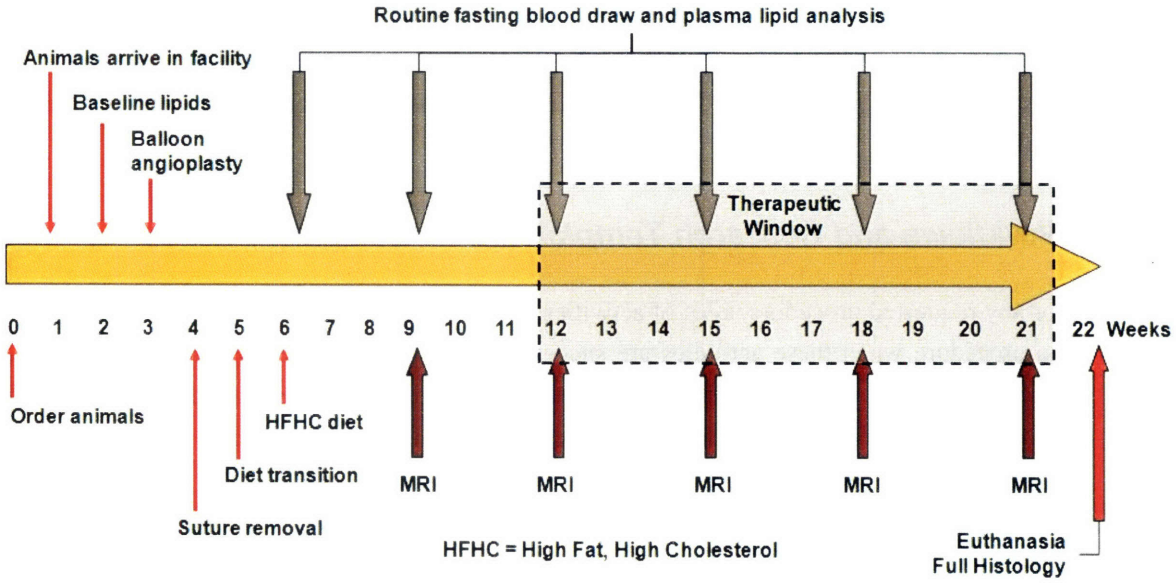


Figure 13- Typical research activities and timeline[18]

[18] Modified from presentation created by Jennifer Allport (NIBR), May 29th 2007

This example illustrates the level of complexity that exists in an in vivo research facility. Hundreds of experiments similar to this example can be underway at any given time. The managers coordinating research operations within the vivariums are required to understand the specifics of each study in order to ensure that the animals are properly conditioned and prepared for future procedures. If any of these steps are not completed according to the experiment design (process and timing), the results for the entire experiment can be invalidated. Coordinating these activities successfully can be a daunting task given the complexity of many research projects. For example, vivarium staff must know that the rabbits discussed in the above example require transition to a new diet (i.e. a different type of food pellet) beginning on the fifth week. This might appear simple, but several simple questions highlight the potential mistakes that can be made if instructions are not communicated accurately:

- What diet should be used during the transition phase?
- Where is the diet obtained from?
- What amount (weight) should be given to the animals?
- Do the animals need to be fed at a specific time of the day?

Communication of these instructions can be tricky even if a good relationship exists between the researcher and technician caring for the animals. These processes become especially difficult if no relationship exists between these individuals or if a technician unfamiliar with the research is tending to the animals. This happens frequently due to sick leave, weekend coverage or vacation coverage. The scheduling tool discussed in this chapter is intended to address these coordination challenges.

2.4. Workflows and Research Templates

For any requested procedure, a set of activities can be defined that will result in the completion of the procedure when these activities are successfully undertaken. The success of the procedure depends on both the timing and order of these activities. In other words, the workflow of activities is critical to the success of the procedure. A workflow is defined as follows:

> ***Workflow:*** The path and systems used in the linked flow of activities with a specific start and finish that describe a process. The flow defines where inputs are initiated, the location of decision points and the alternatives in output paths, and is used in systems that perform automatic routing.[19]

[19] As defined by Bridgefield Group (http://bridgefieldgroup.com/bridgefieldgroup/glos10.htm)

For the atherosclerosis example discussed previously, the workflow for the MRI process would contain the following activities:

1. Transfer Rabbits to the MRI holding area
2. Allow rabbits to acclimate
3. Transfer rabbits to the MRI procedural room
4. Prepare MRI procedural room
5. Collect blood samples and image rabbits
6. Transfer rabbits back to the holding room
7. Transport rabbits back to the main rabbit holding room.

These seven steps are required for just one procedure. The entire research project has a total of 14 procedures. The workflow for this project is created when all activities are linked together in a manner that accurately reflects both timing and activity dependency (i.e. the rabbits must be transported to the MRI prior to imaging). Notice that Steps 3 and 4 can be completed in parallel. By defining activities as workflows, parallel activities can easily be designated. Although workflows for an actual research project can be very complex, the workflow shown in Figure 14 provides a visual example of the MRI workflow discussed above.

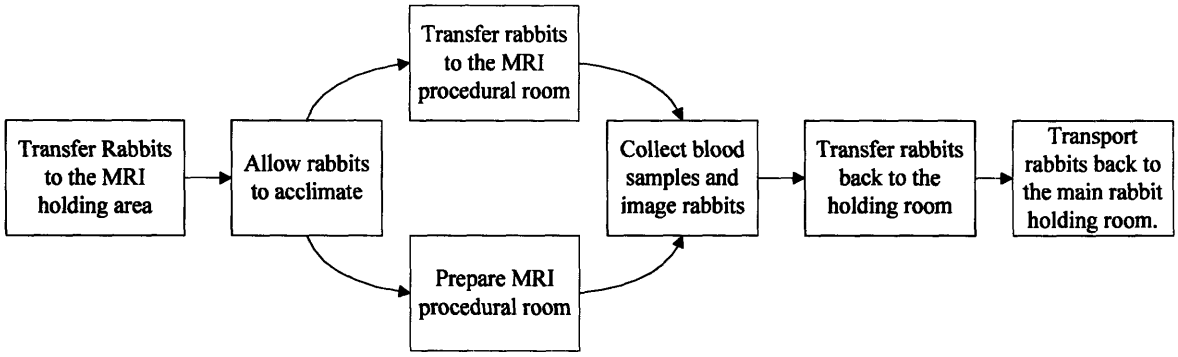


Figure 14- Simple workflow example for rabbit imaging

Developing an efficient process for defining these workflows in a usable format is a key challenge. As mentioned previously, research projects are very dynamic and unique. Scientists typically augment standard procedures with custom activities. A scheduling tool must leverage this commonality, while providing a simple process for customizing a workflow to meet specific research needs. The scheduling process discussed in subsequent sections will utilize standardized workflows called *research templates*, which define the standard activities required to complete individual processes and full projects. These research templates will be defined by a scientist and saved in a shared database. Scientists will be able to save valuable time using research templates since they will not be required to create a completely new workflow when beginning a new

project. In addition, scientists will be able to collaborate more efficiency since the research templates will be available to the in vivo research community.

2.5. Process Integration

The process to plan and execute a research project can be divided into four activities, which are illustrated in Figure 15. The first step involves the creation of research templates. The timing of this step is dependent on the extent to which a scientist understands the research that they plan to perform. The earliest a template can be defined is just after approval of the research protocol, which is typically well in advance of when research is actually conducted in the vivariums. In many cases, the scientists will not fully comprehend the activity detail of the procedures that they would like to perform. Some scientists conduct research that involves new and cutting-edge procedures. Although they might have a basic idea of the activities required to complete these procedures in the protocol definition phase, defining a research template at this stage would be inefficient in some cases. When this is the case, the research templates will be defined much closer to the start of the actual research projects. Creating a research template will require a significant level of process detail. The scientists will most likely need to consult with both their research teams and the vivarium managers in order to develop a functional template. This process could take several weeks and should be started at least a month prior to ordering any animals.

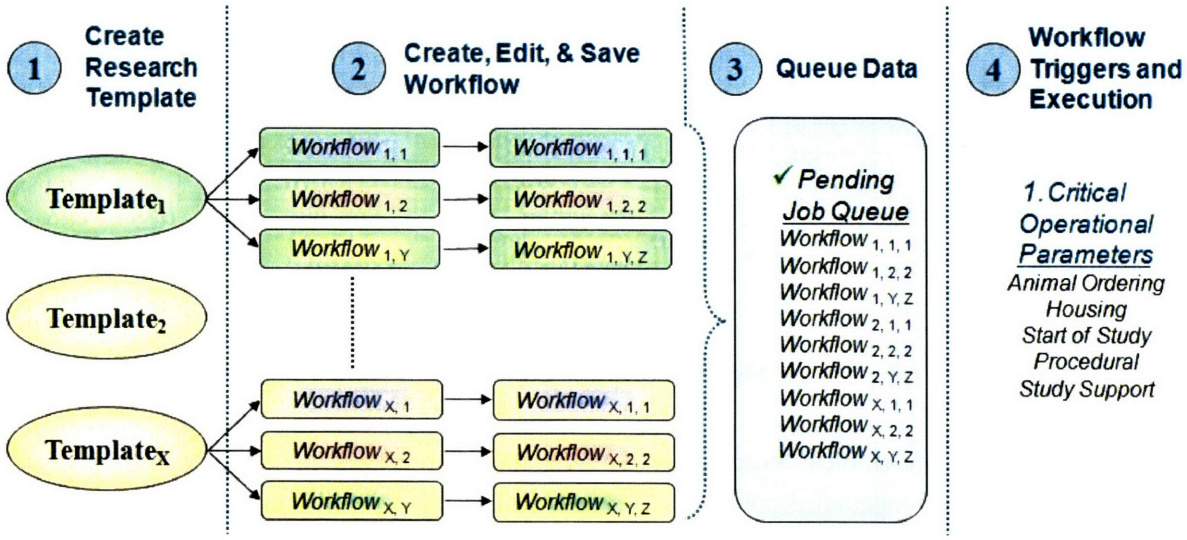


Figure 15- Research scheduling process

The second step involves creating unique workflows for specific research projects. The workflows will be derived from research templates. To create a workflow, a template will be selected from a database. In addition to using a template that had been defined for a particular research project, scientists will have the ability to use a template that they had created for a past

project or a research template from another scientist. Once the template is selected, it is saved as a custom workflow. The scientist then modifies the workflow to add any unique procedural or other activities in order to reflect the specific research that they would like to conduct. Activities can also be deleted from the templates in order to eliminate any unneeded process steps. The subscripts shown in Step 2 of Figure 15 (e.g. $Workflow_{1, Y, Z}$) represent the customization of each workflow. In this example, three unique workflows are being created from one standard research template. Workflow data will feed into the scheduling algorithms that are discussed later in this paper.

Once a workflow is complete, it is moved to the activity queue. While a workflow is in the queue, scientists will have the ability to modify the contents. Scientists should be encouraged to submit workflows into the scheduling tool several weeks in advance of animal ordering since this data can be used to allocate capacity and make operational changes to the research facilities. As activity dates defined in the workflows are approached, the scheduling tool will execute several algorithms used to allocate resources (procedural space, cage/room assignments, etc). For example, the tool will assign procedural space three days in advance of the requested procedural date. Workflows can still be modified after these trigger points have been past. Assignment of resources will be on a first come, first served basis since the system has been optimized and some resources have already been assigned to other projects.

2.6. Tool Development and User Interface

This section describes the attributes and conceptual interface for a web-based scheduling tool. Development of the actual scheduling software is beyond the scope of this project and the framework described in this paper is intended to augment the development of this software. In addition, this scheduling tool has been created assuming that it can be integrated into a more comprehensive vivarium management software package. NIBR is currently in the process of deploying this software package. The scheduling tool must allow scientists to efficiently schedule, monitor and modify their research projects. In addition, vivarium managers will use the tool to view requests, understand capacity issues and coordinate management of ongoing research. The scheduling tool will need to interface with other modules, including protocol development and animal ordering.

The design of the scheduling tool can be accomplished using several approaches. For this analysis, it is assumed that the scientists will have access to a module (external to the scheduling tool) that summarizes important data related to their past, ongoing and planned research. Items included in this view would be pending animal orders, approved protocols, saved research templates and active workflows. This view would be used to manage key aspects of the scientist's research, including research template and workflow creation. The scheduling modules (template creation and workflows) should be assessable from this screen and will have three main components- study information/logistics, workflow detail and special instructions. This

discussion will focus on the workflow data entry screen, since the research template data entry screen contains a subset of this information. A conceptual example of the workflow scheduling module is shown in Figure 16. The data contained in this conceptual model roughly correlates to the atherosclerosis example discussed earlier in this chapter.

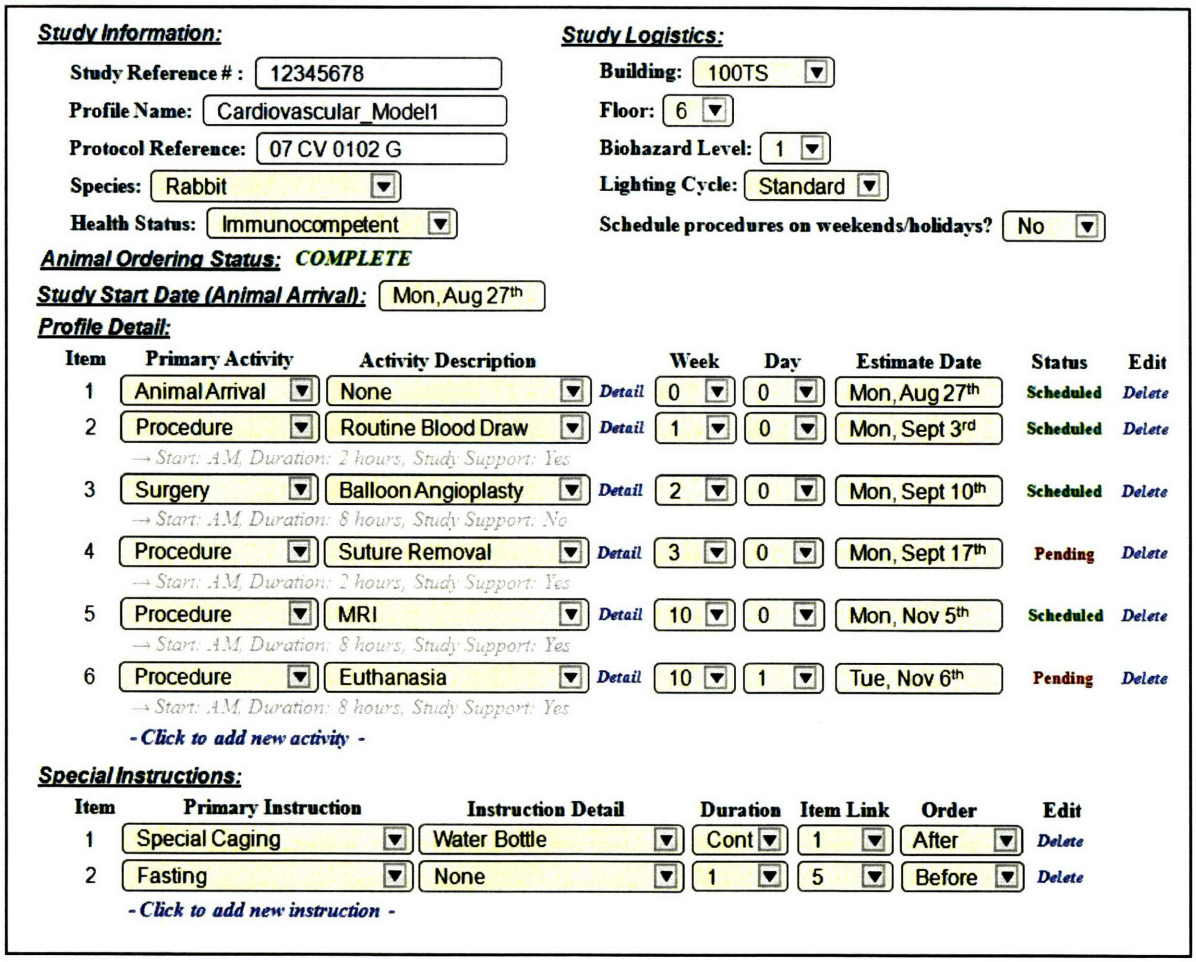


Figure 16- Conceptual workflow scheduling module design

The study information and logistics section of this module is used to link the workflow with critical data that is contained elsewhere in the vivarium management database. Most importantly, the workflow needs to reference the protocol that the research is being conducted under. This will allow extensive rule and constraint checking to occur. Rule checking is needed in order to verify that the requested research does not exceed the number of animals that are available to use per limitations specified in the protocol. In addition, the procedures defined in the workflow section will be developed from the list of approved procedures that is also contained in the protocol. When a new workflow is created, the system automatically assigns a unique *study reference number*. Since multiple species can be included in a protocol, the scientist will have the ability to specify the species and animal health status that will be used. The study logistics

section allows the scientist to indicate their preferred procedural location by specifying the building, floor and specific rooms. Other required information collected in this section includes biohazard level, light cycle preference and weekend scheduling preference. A link to the animal ordering module will also be available in this view.

The workflow detail section allows the scientist to translate their procedural plans into detailed activities that can be used to coordinate their research. Assuming the workflow is being created from an existing template, the activity descriptions will already be populated. If this is the case, the scientist will only need to add or delete a limited number of activities in order to customize the workflow. However, if the scientist is creating a template, the workflow detail section will only have one line item and the scientist will be required to select the *Click to add new activity* option in order to add additional activities to the workflow. As mentioned previously, the activity descriptions will be limited to a list of the activities that have been approved in the protocol. Once an *Activity Description* is selected, the user will have the ability to enter details pertaining to the activity. Details include the requested start time, the duration of the activity and study support requirements. Next, the start date of the activity is entered using the week and day. These values are referenced to the animal arrival date. This approach simulates the research timeline model that was discussed in Section 2.3. When the scientist enters the animal arrival date, the *Estimated Date* section automatically populates. The *Status* field, located to the right of the estimate date field, indicates if the scheduling algorithm has assigned a resource for the listed activity. The status will typically begin as *Pending* and transition to *Scheduled* when scheduling algorithms automatically assign resources. The *Edit* field can be use to delete an entire activity entry. Activities are automatically listed in ascending order using date and time.

Typically, the activities contained in the workflow detail section are coordinated by the scientist and their research team. In addition to these activities, scientists will have special instructions for their study animals. Examples of special instructions include adding water bottles to cages (water is typically delivered with an automated system) or removing food from cages in order to prepare animals for a procedure. Complying with these special instructions is the responsibility of the vivarium management staff. Similar to the workflow detail section, the special instructions are entered using a two step process (primary and detail). The options for these lists are generated from a common set of special instructions and are not dependent on information contained within the protocol. The method used to schedule the special instructions is much different compared to the method used to schedule the workflow activities. Once the special instruction is identified using the drop down lists, the *Item Link* field is used to specify the workflow item number that the special instruction is related to. The *Order* field indicates when the special instruction is performed with respect to the timing of the activity that is it linked with. Finally, the *Duration* field is used to specify the total amount of time for which the special instruction is active. This can be specified in either hours or days. The structure of this format is illustrated with the fasting example shown in Figure 16 (*Item 2* under the special instructions section). As the special instruction indicates, the animals involved in this research will require fasting for one day prior to

the start of "*Item 5*". In this example, *Item 5* is the MRI procedure (as listed in the *Workflow Detail Section*).

2.7. Chapter Summary

Coordinating research activities within a vivarium requires an understanding of the research design in order to develop a process that the research community is willing to adopt. Research activities are typically defined in a linear manner, beginning with the animal order and ending when the animals are euthanized. Discrete procedures are conducted within this time interval. These procedures are separated by periods of time during which animals recover from procedures or are conditioned for future research. Research study designs can be very complex, containing many procedures and special instructions. Development of a web-based scheduling tool has been proposed, which will help organize this information and automate many of the decision processes. Templates and workflows are used to define the research requirements. Once information is entered into this web-based tool, workflows are added into a queue where data is used to assign resources in an optimized fashion. The following chapters discuss two optimization models that can be used to determine the resource allocation strategy.

3. Procedure Space Scheduling Model

3.1. Chapter Overview

This chapter focuses on the development of a comprehensive scheduling algorithm that can be used to coordinate in vivo research within NIBR's vivariums. The research environment described in the first chapter will be explored in a much greater level of detail. This detailed information will help explain and support the scheduling model structure. Research from other industries will be reviewed to understand common scheduling challenges and applicable optimization techniques. Following this literature review, a comprehensive mixed integer linear programming model will be presented. This will begin with a detailed discussion of research parameters and decision variables. Nomenclature for these parameters will be provided. A series of constraints will be presented, along with the rationale driving the need for these constraints. Finally, a multi-criteria objective function is presented. The structure of this objective function is discussed to explain the approach used to optimize allocation of in vivo research assets.

3.2. Procedural Space Allocation and Scheduling

In vivo research conducted at NIBR is very diverse in nature. As a major pharmaceutical company, Novartis invests in research that spans many disease areas. Although the compounds being developed within NIBR are very unique, the methods used to collect data are fairly standard across disease areas. In vivo research at NIBR provides scientists the first opportunity to collect data for a potentially life-saving drug within a living animal. Known as 'models', the animals used for these studies typically have special genetic features that make the animal predisposed to certain diseases. During early in vivo studies, scientists are interested in conducting design of experiments that test many different variables. These activities require large animal populations to generate the required data. In order to make this a financially feasible activity, animal models are required that have low maintenance and housing costs. Rodents such as mice and rats meet these requirements and are well suited for this early stage research. Although slightly large in size, species such as rabbits, ferrets and guinea pigs also make effective research models for certain diseases. As discussed previously, these animals are housed in standardized caging. Cages are typically organized on racks and multiple racks are situated in a holding room.

Many research studies require regimented dosing of compounds into the study animals. The duration of these studies can be as short as several days and as long as a couple years. During this process, animals will typically undergo various procedures to induce a disease response, collect tissue/blood samples from the animals or augment the treatment in some fashion. These can be simple, non-invasive procedures or more complex operations. Some simple procedures can be performed in the holding rooms, while complex procedures typically require specialized

equipment and/or a specialized lab setup. The vivarium facilities at NIBR include many dedicated areas outside the holding rooms that contain this specialized equipment and have lab areas designed specifically for procedural work.

As discussed previously, procedural equipment and facilities are divided into two general categories, flexible and unique. Flexible equipment and procedural space are more generic in nature and are typically more abundant. This includes lab space and actual equipment that can be utilized by multiple research projects and disease areas. An example of flexible equipment is a Biological Safety Cabinet (BSC), many of which are located throughout the NIBR vivariums. A laboratory area that contains a scale, sink and centrifuge is considered flexible procedural space. Unique equipment and procedural space are assets that have limited capacity within the vivarium. An example of a unique procedural space is the rabbit surgical suite, which can only accommodate one research activity at a time.

A fundamentally different approach should be used to schedule these two assets types. Since flexible assets are common throughout the facility, there is an opportunity to use optimization techniques to ensure space is efficiently allocated to the research community. A major focus of this analysis is developing an optimization model that can perform this scheduling function. As will be discussed later in this chapter, there are many constraints and other scheduling requirements. These constraints and requirements complicate the allocation of assets. These assets cannot be effectively managed without the aid of an automated algorithm working in conjunction with a database containing project information and facilities details.

Unlike flexible assets, unique assets can only accommodate a limited number of researchers. This provides little opportunity to optimize the allocation of these assets. For this reason, a different strategy should be used when scheduling equipment and procedural space that fall into this category. A "first come, first served" method allows proactive researchers to schedule the time that they need to conduct their research without using an optimization strategy. Unfortunately, this method discriminates against projects that have short planning horizons or projects that need to be rescheduled, since schedules are immovable and established by individual scientists (difficult to coordinate). A second approach involves ranking projects according to their strategic value to the company. This would require the engagement of a group external to the vivarium management team such as business development. Using either an automated or manual process to schedule these research activities would ensure that the most important jobs get completed first. This method would have the potential to eliminate incentive for researchers to submit timely requests since the schedule would be dictated by the established priority list. Although this analysis does not explore this scheduling strategy in detail, the recommended solution uses a modified version of the project ranking approach where both rank and request order are taken into consideration when scheduling these unique assets. The scheduling model developed later in this section can be extended to include this functionality.

Significant opportunity exists to improve the methods used to allocate research assets within the NIBR vivariums. Today, the process used to allocate these resources is mostly controlled by the individual research groups. This decentralized system is extremely difficult to optimize since decisions are typically made without considering the overall needs of other scientists. When NIBR started operations in 2003, the research groups that existed at the time were allowed to "claim" space within the vivariums. At the time, future development activities of these groups were still being defined. With optimistic and aggressive growth plans, these groups established a research footprint within the vivariums that exceeded their research needs. A mentality quickly developed where these research groups were reluctant to let other research groups utilize their assigned space since they feared this would constrain future growth. Five years later, this system remains in place. The research landscape has changed significantly with the addition of new disease areas and the evolution of research priorities. These new disease areas have difficulty finding space within the vivairums to house animals and perform research. Sufficient capacity exists to accommodate all in vivo research at NIBR, unfortunately this capacity is not being allocated efficiently.

This problem can be solved by implementing a process that assigns resources to scientists based on the overall needs of the organization, not just a single disease area. However, unless the process of managing resources is changed, implementing a system of this nature will be difficult. The current incentives structure for disease areas promotes localized decision making. Although this incentives structure might produce other favorable results, efficient allocation of resources is an undesirable outcome. To combat this problem, one solution involves moving the management of these resources to an unbiased, centralized group. The challenge involved with making this transition is coordinating the various research requests in a manner that is fair, accurate and robust. Scientists have been opposed to this change since a process does not exist that addresses this challenge. The model presented in this section provides a detailed and quantitative process for coordinating these activities.

3.3. Literature Review and Applicable Research

Fundamentally, the problem of coordinating in vivo research is similar to scheduling multiple jobs on machines in a job-shop environment. In this comparison, jobs are the research requests and procedural rooms are equivalent to machines. In addition, holding rooms act as quasi-buffers within a sequence of related research tasks. The problem becomes more complex when the research process is taken into consideration. Typical job shop models and heuristics assume that jobs are placed in queues, from which they are assigned to machines. Many models attempt to minimize system makespan, which is the time difference between the start and finish of a sequence of jobs or tasks. Makespan is not a primary concern when conducting in vivo research since the quality of research data is directly related to the dosing regimen carried out over weeks or months. Instead, timing of the job release and machine assignments are the critical considerations that need to be optimized in an in vivo drug discovery environment.

A literature review of this subject yielded no published research applicable to pharmaceutical R&D scheduling and coordination. A broader search was conducted with the goal of finding models from different industries that can be applied to this unique challenge. Of particular interest was research based on case studies and actual industry challenges. For example, research from the agriculture, steel, airline and healthcare industries was reviewed and assessed for potential application to this scheduling problem found at NIBR.

First, an area of growing interest in the healthcare industry is utilization of operating rooms and recovery space. In this comparison, the animals or cages are similar to patients undergoing surgeries in a hospital environment. NIBR has procedural rooms, which are equivalent to operating rooms at a hospital. Recently, researchers have focused on modeling operating room assignment as a two-stage hybrid flow shop (Jebali et al., 2006; Guinet and Chaabane, 2003). The model developed by Jebali et al. (2006) introduces a two-step approach that first assigns patients to operating rooms, then sequences patient assignments to the operating rooms and recovery beds. The resulting problem is modeled as a mixed integer program with linear constraints and a linear objective function. When comparing this model to the case of in vivo research scheduling, model outputs such as operation/procedural start times, specific room assignments and basic resource alignment are applicable for both models. However, the objective for this operating room scheduling model focuses on managing costs associated with overtime, undertime and patient waiting. Spatial challenges are not considered and patients do not "flow" between their staging rooms and the operating rooms. These latter attributes make the operating room model distinctly different from the in vivo research scheduling challenge.

The second application of interest is the scheduling and coordination of arriving aircraft into terminal gates. Airport operations have been studied in detail due to cost, service and complexity challenges. Several key features of the airport gate assignment problem (AGAP) are applicable to research scheduling at NIBR. Most importantly, the AGAP attempts to minimize the distance passengers (or cargo) travel between gates or between their arriving gate and the airport entrance/exit. In a similar application, vivarium managers attempt to minimize the distance that cages are transported between holding and procedural rooms. This requires knowing the spatial relationship between holding and procedural rooms, as well as the number of cages or animals associated with any given procedure. Like arriving flights on a set schedule, researchers will dictate the preferred time to begin their research. Deviation from this requested time will incur a cost associated with degraded service level.

The basic AGAP is a quadratic assignment problem since flows (unit weights across a unit distance) are measured for location pairs. The AGAP problem has been studied since the late 1960s and was shown by Obata (1979) to be NP-Hard. Dorndorf et al. (2007) provide a particularly good overview of the challenges that airport managers face and the techniques that have been used to address these challenges. Recently, heuristic models have been developed that provide near optimum solutions for the AGAP problem. Although these models do not guarantee

an exact solution, data suggests that these heuristic models require substantially less processing time compared to competing methods that use branch-and-bound LP solver solutions (e.g. CPLEX[20]). In addition, solutions typically converge to near optimum solutions.

Xu and Bailey (2001) developed one of the first heuristic models to address customer connection times as they relate to the AGAP. This solution begins with a quadratic objective function, but uses a reformulation technique to develop a mixed integer programming model with both linear constraints and a linear objective function. A tabu-search meta-heuristic is developed that uses special neighborhood moves and candidate list strategies that combine to produce an effective algorithm for solving the AGAP. This model is leveraged by Lim et al. (2005), who developed a model that minimizes transfer costs between connecting flights (passengers or cargo), while accounting for discrepancies in flight arrival and departure times. This model also uses a tabu-search strategy that leverages insert and interval exchange moves together with a time shift algorithm. A solution for over-constrained flight to gate assignments is proposed by Ding et al. (2004), which accounts for situations where arriving flights outnumber available gates. In these situations, airplanes are required to load and unload passengers on an area of the airport tarmac called the apron. A greedy algorithm helps minimize the number of flights that must use the apron areas since doing so adversely impacts customer service. An integer program with multiple objectives has been developed by Drexl and Nikulin (2008). This model employs a three part objective function to minimize unassigned flights and passenger walking distance, while maximizing flight gate assignment preference. Pareto simulation annealing is used to find optimum solutions since this model has quadratic constraints and a quadratic objective function.

The model developed in this chapter is a mixed integer linear programming formulation. Aspects of this model are rooted in the methods that have been employed by Xu and Bailey (2001) and Lim et al. (2005). Although some important similarities exist between the airline and pharmaceutical industries, the concepts and variables have been adapted to reflect the unique challenges of the RAP. Specifically, additional constraints and a unique objective function lead to a formulation that is unique in literature.

3.4. Parameters

This section details all parameter inputs for the RAP model. All data inputs are assumed to be deterministic. In addition, all time values are treated as the discrete points in a time horizon.

[20] CPLEX is a mathematical optimization tool developed by ILOG, Inc. http://www.ilog.com/

3.4.1. Research Demand

In general, research activities in a vivarium are coordinated by the vivarium manager. The goals of a vivarium manager are twofold. First, the vivarium manager must ensure that the animals in their facilities receive a level of care that is at or above regulatory standards (USDA, AAALAC, etc.). Their second goal is to ensure that research can be conducted in a robust and efficient manner. The scientists will ultimately be responsible for executing the research, but the vivarium managers are responsible for providing an environment in which this can occur.

Drug discovery is a very fickle process. Scientists are constantly evaluating research data and adjusting their research plans to reflecting changing hypotheses. It is critical to have a vivarium and scheduling process that can accommodate this variability. At NIBR, a scheduling process will not be successful unless scientists are given the opportunity to request the day and time that they would like to conduct procedures. For this reason, we expect to see a clustering of requests around days and times that are convenient for scientists to complete their work. Interviews with scientists indicate they prefer to begin procedures in the morning to avoid working into the evening. Typically vivarium staff begin their work shift at 7 a.m. and leave the facility in mid afternoon. In some instances, the timing of research is driven by the interaction of the compound with the animal. The scheduling model developed in this paper accounts for these behaviors and research requirements.

In order to determine research demand, the scheduling process will use a web-based interface to gather research requests from scientists (discussed in the previous chapter). When gathering these requests, no limitations will be provided as to total number of jobs or total research hours allowed for a given day. These requests will be scheduled according to the optimization rules, which will attempt to accommodate all requests for the specified day and time. The term *jobs* will refer to the research requests that are submitted by scientists. The following nomenclature will be used to denote daily job requests:

$\boldsymbol{J}$ The set of all jobs that are requested to be completed on a given day.

3.4.2. Procedure Rooms and Procedural Space

Typically, procedure rooms are distinct areas within a vivarium where scientists perform both general and specialized operations. Rooms designated for general use are typically sized between 100-300 ft^2, contain a sink and have bench top space that can accommodate one or more research teams. These rooms can contain specialized equipment. Rooms that are dedicated to a small number of functions typically contain equipment or other apparatus that limit the functionality of this space to the function of this equipment. Although it is possible to keep animals in these areas overnight, this is not typically practiced and this function is reserved for holding rooms.

Non-invasive procedures are also performed within holding rooms. A non-invasive procedure is one that does not place an animal under extensive stress or pain. Scientists prefer to perform these

simple procedures in the holding rooms since this procedure space is in close proximity to their animals. This requires less time to transfer animals, which can be significant when performing procedures on a large number of animals. Procedures cannot occur in the holding rooms if the body cavity of the animal will be opened, the animal will emit loud/prolonged sounds, or if the procedure is terminal (i.e. death).

The varied size and function of procedural rooms complicates the scheduling process since some rooms can accommodate more than one research group, while other rooms cannot. The scheduling system needs to have the ability to place multiple research teams in a single procedural room when the research is compatible (i.e. species, equipment) and when there is space available. For example, research teams typically work in holding rooms at the same time. Not allowing these teams to share the space would be detrimental to overall research efficiency. To accommodate this need, the scheduling heuristic will focus on allocating procedural space, versus procedural rooms. Although this approach does not allow for dynamic capacity allocation and optimization, focusing on procedural space does achieve the larger goal of multiple research appointments to a single room. In general, the vivarium manager will designate the procedural rooms and the corresponding procedural spaces for a given room. In some instances, the procedural space will be the procedure room. In other cases, multiple procedural spaces will exist in one procedure room. The following notation will be used to designate procedure space:

P The set of all vivarium procedural spaces at a given research site. These spaces can be located on multiple floors and in different buildings.

Although this nomenclature uses consecutive numbers to identify the rooms, which does not align with standard architectural room numbering, a decoder can be developed that correlates these two numbering conventions to one another.

A relationship is also required to couple common procedural spaces and ensure that species interactions are scheduled in accordance with government regulations and operational requirements. Typically these relationships are based on procedural space sharing a common room and will be defined by the vivarium manager.

$$X_{il} = \begin{cases} 1, & \text{if procedure space } i \in P \text{ shares the same room as procedure space } l \in P \\ 0, & \text{Otherwise} \end{cases}$$

3.4.3. Holding Rooms

As discussed previously, the primary function of holding rooms is to house animals while these animals are involved in research. Since NIBR is a drug discovery organization, scientists typically use mouse, rat and rabbit models. These animals are kept in cages that are placed on a rack. NIBR policy states that a single cage can house up to four mice or two rats. Rabbits are kept in individual cages. The layout, size and capacity of these rooms can differ significantly

based on species and type of research. In general, it is less difficult to reconfigure holding rooms as compared to procedural rooms.

When developing a scheduling tool for vivarium operations, it is important to allow for simple reconfiguration of the space attributes within the tool. This is required since research can fluctuate, requiring vivarium managers to re-optimize space. In general, the vivarium manager will assign holding rooms and their respective attributes. The following notation will be used to designate holding rooms:

$\boldsymbol{H}$ The set of all holding rooms at a given research site. These rooms can be located on multiple floors in different buildings.

3.4.4. Cages

The size of research jobs can differ significantly. Some jobs will only require the transport of several cages, while other jobs involve transporting over a hundred cages to a procedural space. The size of each job, as measured by the number of cages involved, must be understood when allocating procedural space. Jobs with a greater quantity of cages will be given preference in the optimization model since transporting these cages will be more burdensome for the scientist. The following notation will be used to designate the number of cages:

$\boldsymbol{c_j}$ Number of cages associated with job $j \in J$, where c_j is a positive integer

3.4.5. Species

The variety of research conducted at NIBR requires the use of many different animal models in order to stay on the cutting edge of science. Scientists are constantly targeting new disease mechanisms. The research community at NIBR develops these models internally, as well as with external scientists from academia and other companies. Most scientists tend to use rodent models (mice and rats) since the vast majority of research conducted in Cambridge is drug discovery. Other species found in the Cambridge vivariums include rabbits, cotton rats, ferrets, and hamsters.

$\boldsymbol{B}$ The set of all distinct species used for research at a given research site.

Individual jobs are limited to one species. Therefore, a distinct species can be associated with each job using the following notation:

$$\boldsymbol{V_{jb}} = \begin{cases} 1, & \text{if species } b \in B \text{ is associated with job } j \in J \\ 0, & \text{Otherwise} \end{cases}$$

3.4.6. Procedure Space Species Restrictions

Government regulations require procedures involving different species to be conducted in separate rooms. It is critical to understand what procedures are being conducted in a room in order to ensure that there is no species overlap. In addition, some procedure rooms will have species restrictions that limit the type of species that can enter a room. For example, when a holding room functions as a procedure room, only species identical to those housed in the room can utilize this area for procedures. An array will be used to specify alignment of species to procedural spaces.

$$S_{bi} = \begin{cases} 1, & \text{if species } b \in B \text{ is allowed to be processed in procedure space } i \in P \\ 0, & \text{Otherwise} \end{cases}$$

3.4.7. Procedure Space Preference

Typically, scientists select the procedure space in which they would prefer to conduct their research. The current operating model has optimum flexibility for scientists that have dedicated space since they "own" the rooms and are typically not required to coordinate with other scientists. This new scheduling approach will require a greater level of coordination since these rooms will be available for all scientists to use. The objective function will be structured in a way to help accommodate the scientist's space preference. Accommodating these requests will not be a hard requirement and room conflicts will be resolved using several optimization criteria. A scheduling tool will have the ability to collect feedback from scientists regarding their room preferences. The following notation will be used to denote this preference:

$$Y_{ji} = \begin{cases} 1, & \text{if procedure room } i \in P \text{ is selected as the preferred location for job } j \in J \\ 0, & \text{Otherwise} \end{cases}$$

Scientists will have the ability to designate more than one procedural space as "preferred".

3.4.8. Room Proximity and Logistical Impact

In vivo research operations differ significantly in scope and scale across companies and across research sites. Over 1000 scientists actively conduct research in the NIBR's Cambridge facilities, a significant portion of which is in vivo research. To develop a flexible scheduling system, this system must consider spatial dependencies. In addition to keeping their animals in a consolidated footprint (i.e. a small number of holding rooms), scientists typically prefer to use procedural space that is in close proximity to their animals. Scientists also prefer to use the same procedural rooms for their research in order to become comfortable with the nuances of these rooms. To accommodate these preferences, the scheduling tool must take into account the scientist's room preference, as well as the spatial relationships between the holding room(s) containing a scientist's animals, the preferred procedural space and available procedural space.

A fully optimized schedule would place all research in the rooms that have been requested by the scientists ($Y_{ji} = 1$). At times this will not be possible due to research conflicts and space constraints. When this occurs, the optimization model will attempt to minimize the logistical impact between the animal holding room and the procedure space that is selected. To understand these spatial relationships, the vivarium manager will create a matrix in the scheduling tool that defines the logistical impact for all procedural spaces in the set P. The initialization of this data will be time consuming, since architectural data will need to be used to determine these relationships. This will only need to be completed once since vivarium layouts typically do not change. The following notation will be used to specify the spatial relationship between procedure spaces and holding rooms:

Z_{ji} The logistical impact between the holding room containing animals for job $j \in J$ and procedure room $i \in P$, where Z_{ji} is a positive integer

Consideration needs to be given to rooms that are located on different floors and in different buildings. Biosecurity best practices strongly recommend against the movement of animals from the floor that they are housed on. Trafficking of animals between floors and buildings increases the likelihood of spreading pathogens from one area in the vivarium to another. When pathogens are found within a facility, the vivarium manager typically quarantines the contaminated area to prevent further spread of the pathogens. Quarantine requires scientists to ramp down or terminate their research in the affected area. This can cost millions of dollars in operational expenses and lost research when a large portion of the facility is impacted.

Several simplifications can be made to help streamline the process of building the logistical impact matrix. In general, the impact of moving between floors is roughly the same. It is not more or less difficult to move animals from floor A to floor B, as it is to move from floor A to floor C. The same guideline can be applied to building transfers. Once an animal leaves a building, transferring the animal to another building will take approximately the same amount of effort and incur the same biosecurity risks. The impact of moving between floors is typically lower than the impact of moving between buildings. If these assumptions hold true, a vivarium manager can use estimates for the floor-to-floor and building-to-building transfer logistical impacts to develop the Z_{ji} matrix. The following notation defines the logistical impact for animal moves that leave a given floor:

f The logistical impact of moving animals between floors, but within the same facility

g The logistical impact of moving animals between facilities

In general, these logistical impact measurements are determined by the vivarium manager and are established relative to the logistical impact of intra-floor animal transfers. For this analysis, the magnitude of the logistical impact contributed to intra-floor animal transfers will use the following notation:

Q_{hi} The logistical impact of moving animals between holding room $h \in H$ and procedure room $i \in P$, both located on the same floor. For simplicity, this analysis assumes Q_{hi} is equal to the linear distance required to travel between these two rooms.

Variables f, g and Q_{hi} are used to create the Z_{ji} matrix. For example, assume that a small facility has four procedure rooms ($P = \{A, B, C, D\}$). Vivarium managers would like to understand the logistical impact of scheduling a certain job j into one of these four rooms. The animals for job j are located in holding room h. Rooms A and B are located on the same floor as h. Room C is located on a different floor from h, but is still in the same building. Room D is located in a different building. Also assume that $Q_{hA} = 100$ (feet), $Q_{hB} = 200$ (feet), $f = 500$ (feet equivalent) and $g = 1000$ (feet equivalent). Using these values, the Z_{ji} matrix is defined as follows:

$$Z_{ji} = \begin{matrix} A \\ B \\ C \\ D \end{matrix} \begin{bmatrix} Q_{hA} \\ Q_{hB} \\ f \\ g \end{bmatrix} \quad \Rightarrow \quad Z_{ji} = \begin{bmatrix} 100 \\ 200 \\ 500 \\ 1000 \end{bmatrix}$$

3.4.9. Equipment Attributes

A wide array of equipment and facility infrastructure is required to support in vivo research at NIBR's Cambridge campus. The vast majority of this equipment and infrastructure are located within the vivarium. Equipment can be categorized into two groups- unique and common. Unique equipment is limited in availability. For example, there is only one surgical area that can accommodate an animal the size of a rabbit. Common equipment is widely available within the vivarium. This analysis is made easier by including facility infrastructure in this category. Facility infrastructure is integral to the building and cannot easily be moved to another area of the vivarium. Fume hoods and biological safety cabinets (BSC) are examples of facilities infrastructure that scientists might require to conduct their research. The following notation will be used to identify these items:

E The set of all unique equipment and facilities infrastructure that is found within the vivariums at NIBR's Cambridge campus.

3.4.10. Procedure Space Equipment Alignment

Procedure space allocation will be partially dependent on equipment and facilities that are required to successfully complete the specified research. To be effective, the scheduling process must assign research requests to procedure space that contain the required equipment. For example, scientists might require access to an ultrasound machine or surgical table. The room that is assigned to their research must contain this equipment. Similar to species restrictions, an array will be used to correlate equipment and infrastructure with procedural space. A particular procedural space can support more than one equipment and/or infrastructure component (e.g. a

single procedural space contains both an ultrasound machine and BSC). This matrix uses the following nomenclature:

$$W_{ei} = \begin{cases} 1, & \text{if equipment } e \in E \text{ is contained within procedural space } i \in P \\ 0, & \text{Otherwise} \end{cases}$$

Next, a parameter is required to capture the requirements of the scientist performing a specific job. When submitting a work request, a scientist will have the opportunity to select the equipment that is required either from a pre-populated profile or from a drop-down list of available equipment (equivalent to E). The following notation will be used for this parameter:

$$N_{je} = \begin{cases} 1, & \text{if equipment } e \in E \text{ is required to perform job } j \in J \\ 0, & \text{Otherwise} \end{cases}$$

3.4.11. Hours of Operation

Although scientists have 24 hour access to the vivarium, research is rarely conducted during the evening and night hours. Scheduling research during these times would not be supported by the in vivo research community. For this reason, upper and lower bounds need to be established for research scheduling. These bounds are defined as:

γ Lower scheduling bound- the earliest research is scheduled to begin

λ Upper scheduling bound- the latest research is scheduled to be completed.

Typical hours of operation are from 8 a.m. to 4 p.m. To allow for greater flexibility, the scheduling bounds will extend beyond these standard hours of operation. This analysis will assume that the first jobs can be scheduled as early as 6 a.m. and must be completed by 6 p.m.

3.4.12. Researcher Start Time Preference

The intent of this scheduling tool is to assign specific times when researchers can begin their in vivo research. The optimal start time will be defined by the scientist and will be collected using the web-based scheduling tool. The following notation will be used to identify this data:

ω_j The preferred start time for job $j \in J$, where ω_j is a positive integer in the range $\lambda \leq \omega_j \leq \lambda$

Research scheduling will optimize tradeoffs between procedural space preference and procedure start time. Scientists place different priorities on these two variables. Some scientists feel that their research needs to begin at precise times (metabolism compounds for instance), while other scientists place a higher priority on conducting research in their preferred procedural space. To accommodate these contrasting preferences, a prioritization variable will be used to place emphasis on one of the two variables:

$$\phi_j = \begin{cases} 1, & \text{if job } j \in J \text{ is prioritized based on the preferred start time} \\ 0, & \text{if job } j \in J \text{ is prioritized based on the preferred room assignment} \end{cases}$$

3.4.13. Research Duration

Research duration, also known as processing time, is defined as the total amount of time required to transport animals, setup equipment, perform the necessary research, return animals to their housing room and return the space to its original condition. Accurately predicting the research duration is important since an inaccurate predication will impact subsequent research. Two methods can be used to determine the research duration. First, standard times can be calculated using historical data or estimates from research experts. With standard times, formulas can be entered into a tool that calculates the research duration based on the research being performed, the number of animals involved and logistical considerations. Alternatively, the researchers can be asked to provide an estimate of the research duration.

Both options have risks and benefits. Standard times would provide the most consistent estimates if the research being performed was routine and consistent in nature. Unfortunately, this assumption does not apply in a significant percentage of cases. Allowing researchers to specify the research duration provides other challenges. Without any incentives to curtail the amount of time reserved for procedures, scientists can request more time than they actually need to perform their research. This affords the scientist extra buffer time that prevents full optimization of the research schedule. However, allowing researchers to specify their time is much easier than establishing standard processing times for many different procedures. For this reason, it is recommended that the research duration be defined by the scientist performing the research. Vivarium managers should validate the accuracy of data being entered into the scheduling tool and take appropriate action if discrepancies are found. The research duration is defined as:

Δ_j The estimated research duration (processing time) for job $j \in J$

3.4.14. Weighting Factors

The following parameter will be used to customize penalty weightings within the scheduling model's objective function:

α Weighting factor used to balance and amplify components of the objective function

Ψ A sufficiently large number

3.4.15. Parameter Summary

Refer to Table 1 for a summary of the procedural space scheduling model parameters:

Table 1- Parameter Overview

Parameter	*Description*
B	The set of all distinct species used for research at a given research site
c_j	The number of cages associated with job $j \in J$
E	The set of all unique equipment and facilities infrastructure
f	The logistical impact of moving animals between floors (same facility)
g	The logistical impact of moving animals between facilities
H	The set of all holding rooms at a given research site
J	The set of all jobs that are requested on a given day
N_{je}	The set of equipment required to perform job j
P	The set of all vivarium procedure spaces at a given research site
Q_{hi}	The logistical impact of moving animals between holding and procedure rooms
S_{bi}	Procedure space species restriction variable
V_{jb}	Job to species alignment parameter
W_{ei}	Procedure space to equipment alignment parameter
X_{il}	Procedure space linkage parameter (for shared procedural rooms)
Y_{ji}	Procedure room preference parameter
Z_{ji}	Logistical impact parameter
α	Objective function weighting factor
Δ_j	The estimated research duration (processing time) for each job
γ	Lower scheduling bound- the earliest time research is scheduled to begin
λ	Upper scheduling bound- the latest time research is scheduled to be completed
ϕ_j	Research start time preference indicator
ω_j	The preferred job start time as defined by the scientist
Ψ	A sufficiently large number

3.5. Decision Variables

3.5.1. Procedure Start Time

This model has two primary decision variables- procedure start time and room assignments. The procedure start time will determine when scientists are given access to their designated procedural space in the vivarium and is denoted as:

t_j The assigned start time for job $j \in J$

3.5.2. Procedural Space Assignments

A binary approach will be used to assign research requests to procedural space. This decision variable has the following nomenclature:

$$R_{ji} = \begin{cases} 1, & \text{if job } j \in J \text{ is assigned to procedure space } i \in P \\ 0, & \text{Otherwise} \end{cases}$$

3.5.3. Procedure Space Staging

The procedural space staging variable helps determine the order in which research is scheduled when multiple jobs share the same procedural space and is denoted as:

$$M_{jk} = \begin{cases} 1, & \text{if job } j \in J \text{ is completed prior to the start of job } k \in J \\ 0, & \text{Otherwise} \end{cases}$$

In addition to understanding the relationship between jobs that share the same procedural space, an additional binary variable is required to define the relationship between jobs that are assigned different procedural spaces. This decision variable has the following nomenclature:

$$A_{jkil} = \begin{cases} 1, & \text{if job } j \in J \text{ is assigned to procedure space } i \in P \text{ and job } k \in J \text{ is assigned to procedure space } l \in P \\ 0, & \text{Otherwise} \end{cases}$$

3.5.4. Decision Variable Summary

Refer to Table 2 for a summary of the procedural space scheduling decision variables:

Table 2- Decision Variables

Parameter	*Description*
A_{jkil}	Procedure space coupling variable
M_{jk}	Procedure space staging variable
R_{ji}	Procedural space assignment variable (request to space)
t_j	Assigned procedure start time

3.6. Constraints

3.6.1. Single Procedural Space Assignment

The scheduling strategy does not allow a single research request to be split across different research spaces. If a scientists has the need for two spaces, these spaces will be reserved separately through the scheduling tool. This constraint can be enforced by ensuring that each job is only assigned to exactly one room:

$$\sum_{i=1}^{P} R_{ji} = 1, \quad 1 \leq j \leq J \tag{1}$$

Equation 1- Variable Definition Review

Decision Variables:

R_{ji}	Procedural space assignment variables (1 if job $j \in J$ is assigned to procedural space $i \in P$)

Parameters:

P	The set of all vivarium procedure spaces at a given research site
j	Subscript used to denote a specific job $j \in J$
i	Subscript used to denote a specific procedure space $i \in P$

The converse relationship does not hold true since a single procedure space can accommodate more than one research request.

3.6.2. Coupling Variable

The coupling variable A_{jkil} is defined with respect to the procedural space assignment variable. Three equations establish this relationship. First, the coupling variable will never be greater than either of the two components that establish the value of this variable. For example, if a certain job $j \in J$ is not assigned to procedure space $i \in P$, then the coupling variable by definition will be assigned a value of zero. A similar argument is made for job $k \in J$ and procedure space $l \in P$. These relationships are written as:

$$A_{jkil} \leq R_{ji}, \quad 1 \leq j,k \leq J, 1 \leq i,l \leq P \tag{2}$$

$$A_{jkil} \leq R_{kl}, \quad 1 \leq j,k \leq J, 1 \leq i,l \leq P \tag{3}$$

These two equations establish the upper limit for the coupling variable. The lower limit can be defined as follows:

$$A_{jkil} \geq R_{ji} + R_{kl} - 1, \quad 1 \leq j,k \leq J, 1 \leq i,l \leq P \tag{4}$$

Equations 2, 3, 4- Variable Definition Review	
Decision Variables:	
A_{jkil}	Procedure space coupling variable (1 if job $j \in J$ *is* assigned to space $i \in P$ and job $k \in J$ assigned to space $l \in P$)
R_{ji} , R_{kl}	Procedural space assignment variables (1 if job $j \in J$ or $k \in J$ is assigned to procedural space $i \in P$ or $l \in P$, respectively)
Parameters:	
j, k	Subscripts used to denote a specific job $j \in J$, $k \in J$
i, l	Subscripts used to denote a specific procedure space $i \in P$, $l \in P$

3.6.3. Time Bounds

The current vivarium operation is only supported during standard laboratory working hours. Scientists have the ability to perform procedures during non-standard times, but this rarely happens. Two equations are required in order to ensure that job requests are not scheduled outside of these time bounds. The first equation defines the earliest a job can begin in the morning and is written as:

$$t_j \geq \gamma, \quad 1 \leq j \leq J \tag{5}$$

A typical value for γ is 6 a.m. This allows scientists to prepare their animals for procedures that are time sensitive and must begin in the early morning. The evening bound is aligned with the research completion time and is written as:

$$t_j + \Delta_j \leq \lambda, \quad 1 \leq j \leq J \tag{6}$$

A typical value for λ is 6 p.m. Although the scheduling tool will allow the vivarium manager to adjust the hours of operation, setting scheduling time constraints builds rigidity into the scheduling system. Requests to perform procedures outside these hours of operations will need to be handled as special requests with the vivarium manager.

Equations 5, 6- Variable Definition Review	
Decision Variables:	
t_j	Assigned procedure start time for job $j \in J$
Parameters:	
Δ_j	The estimated research duration (processing time) for job $j \in J$
γ	Lower scheduling bound- the earliest research is scheduled to begin
λ	Upper scheduling bound- the latest research is scheduled to be completed
j	Subscript used to denote a specific job $j \in J$

3.6.4. Shared Room Restrictions

Several constraints are required to prevent "double booking" of procedural space. These constraints ensure a single procedural space is not occupied by different jobs at overlapping times. Relationships must be established between the job start time, research duration, the procedural space staging variable and the coupling variable. To begin, an equation is developed that ensures the shared procedural space variable will properly identify a situation where two jobs do not overlap:

$$(t_j + \Delta_j) - t_k + M_{jk}\Psi > 0, \quad 1 \leq j,k \leq J \tag{7}$$

In this equation, M_{jk} must equal 1 when $t_k \geq t_j + \Delta_j$, which occurs when job j is completed prior to the start of job k (no overlap). Notice that M_{jk} can be assigned a value of 0 or 1 if the two jobs overlap. To account for this, another equation is developed to address the case of job overlap:

$$(t_j + \Delta_j) - t_k - (1 - M_{jk})\Psi \leq 0, \quad 1 \leq j,k \leq J \tag{8}$$

This equation indicates that M_{jk} must equal 0 when $t_j + \Delta_j > t_k$, which occurs when job k is started prior to the completion of job j. Notice that M_{jk} can be assigned a value of 0 or 1 if the two jobs do not overlap. A third constraint is needed to ensure that one procedural space is not used by two jobs simultaneously. This is accomplished using the following equation:

$$M_{jk} + M_{kj} \geq A_{jkii}, \quad 1 \leq j,k \leq J, j \neq k, 1 \leq i \leq P \tag{9}$$

Notice that either job j needs to be completed prior to job k starting or vice versa in order for the coupling variable to indicate these two jobs share the same procedural space. This achieves the desired outcome. Finally, a constraint is required to ensure proper sequencing of jobs:

$$\boldsymbol{M_{jk} + M_{kj} \leq 1}, \quad 1 \leq j, k \leq J, j \neq k \tag{10}$$

This constraint allows job j to precede job k ($\boldsymbol{M_{jk} = 1}$), job k to precede job j ($\boldsymbol{M_{kj} = 1}$), or for an overlap to occur ($\boldsymbol{M_{jk}}$, $\boldsymbol{M_{kj} = 0}$). However, it does not allow both job j and job k to precede each other.

Equations 7, 8, 9, 10- Variable Definition Review

Decision Variables:

$\boldsymbol{t_j, t_k}$	Assigned procedure start time for jobs $j \in J$ *and* $k \in J$
$\boldsymbol{A_{jkii}}$	Procedure space coupling variable (1 if job $j \in J$ *is* assigned to procedural space $i \in P$ and job $k \in J$ is assigned to procedural space $i \in P$)
$\boldsymbol{M_{jk}}$	Procedure space staging variable (1 if job $j \in J$ is completed prior to the start of job $k \in J$)
$\boldsymbol{M_{kj}}$	Procedure space staging variable (1 if job $k \in J$ is completed prior to the start of job $j \in J$)

Parameters:

$\boldsymbol{\Delta_j}$	The estimated research duration (processing time) for job $j \in J$
$\boldsymbol{\gamma}$	Lower scheduling bound- the earliest research is scheduled to begin
$\boldsymbol{\lambda}$	Upper scheduling bound- the latest research is scheduled to be completed
$\boldsymbol{j, k}$	Subscripts used to denote a specific job $j \in J$, $k \in J$
$\boldsymbol{i}$	Subscript used to denote a specific procedure space $i \in P$

3.6.5. Species Restrictions and Interactions

Government regulations and operational requirements create several key constraints that prevent the interaction of species. As discussed in Section 3.4.6, individual procedure spaces will typically be restricted to a subset of the total species that are housed within the vivarium. To incorporate these restrictions into the model, the following constraint is employed:

$$\boldsymbol{R_{ji} \leq \sum_{b=1}^{B} V_{jb} S_{bi}} \quad 1 \leq i \leq P, 1 \leq j \leq J \tag{11}$$

Equation 11- Variable Definition Review	
Decision Variables:	
R_{ji}	Procedural space assignment variable (1 if job $j \in J$ is assigned to procedural space $i \in P$)
Parameters:	
S_{bi}	Procedure space species restriction (1 if species $b \in B$ is allowed to be processed in procedure space $i \in P$)
V_{jb}	Job to species alignment (1 if species $b \in B$ is associated with job $j \in J$)
b	Subscript used to denote a specific animal species $b \in B$
j	Subscript used to denote a specific job $j \in J$
i	Subscript used to denote a specific procedure space $i \in P$

The following example provides insight into this relationship. Assume that four jobs require scheduling. The species attributes for these four jobs are shown below in Table 3:

Table 3 - Species to job alignment example

Job (*j*)	Species Designation (*b*)	Species Name
1	1	Mouse
2	2	Rat
3	1	Mouse
4	3	Rabbit

Using the information in Table 3, the V_{jb} matrix can be defined as:

$$V_{jb} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix} \end{array}$$

3 = Species Designation (b)

↑ Job Designation (j)

Also, assume the procedural space species restriction matrix S_{bi} is defined as:

$$\mathbf{S_{bi}} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix} \end{array}$$

4 = Procedure Space (i)

↑ Species Designation (b)

The above matrix indicates that mice are allowed to be processed in procedural spaces 1 and 2, rats are allowed to be processed in procedural spaces 1 and 3, and rabbits can only be processed in procedural space 4. When the quantity $\sum_{b=1}^{B} V_{jb}S_{bi}$ is calculated for each combination of i and j, the following constraint matrix is obtained:

$$\mathbf{R_{ji}} \leq \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 = \text{Procedure Space } (i) \end{array} \\ \begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix} \\ \uparrow \text{ Job Designation } (j) \end{array}$$

A second species related constraint is required to ensure that different species are not processed in adjacent spaces (i.e. in the same procedural room) at overlapping times. This constraint requires the species to be identical when both the schedules and procedure room assignments overlap:

$$X_{il}\left(\sum_{b=1}^{B}|V_{jb} - V_{kb}|\right)(A_{jkil} - M_{jk} - M_{kj} + 1) \leq \sum_{b=1}^{B}|V_{jb} - V_{kb}| \tag{12}$$

$$1 \leq j,k \leq J\,, j \neq k\,, 1 \leq i,l \leq P$$

Species are identical when the term $\sum_{b=1}^{B}|V_{jb} - V_{kb}|$ equals 0. When species are different, this same quantity will equal 2. This is true since any job is limited to one species. The expression $\sum_{b=1}^{B}|V_{jb} - V_{kb}|$ will always equal zero (i.e. same species) when the following three relationships are true:

1. $X_{il} = 1$, meaning that the procedural spaces i and l are in the same procedure room.
2. $A_{jkil} = 1$, meaning that job j is assigned to procedure space i and job k is assigned to procedure space l.
3. M_{jk}, $M_{kj} = 0$, meaning the schedules for job j and job k overlap.

When these three conditions are met, the inequality reduces to

$$2\sum_{b=1}^{B}|V_{jb} - V_{kb}| \leq \sum_{b=1}^{B}|V_{jb} - V_{kb}| \tag{13}$$

which requires the quantity $\sum_{b=1}^{B}|V_{jb} - V_{kb}|$ to equal zero.

Equation 12- Variable Definition Review	
Decision Variables:	
A_{jkil}	Procedure space coupling variable (1 if job $j \in J$ *is* assigned to procedural space $i \in P$ and job $k \in J$ assigned to procedural space $l \in P$)
M_{jk}	Procedure space staging variable (1 if job $j \in J$ is completed prior to the start of job $k \in J$)
Parameters:	
V_{jb}, V_{kb}	Job to species alignment (1 if species $b \in B$ is associated with job $j \in J$ *or* $k \in J$, respectively)
X_{il}	Procedure space linkage parameter (1 if procedure space $i \in P$ shares the same room as procedure space $l \in P$)
b	Subscript used to denote a specific animal species $b \in B$
j, k	Subscripts used to denote a specific job $j \in J$, $k \in J$
i, l	Subscripts used to denote a specific procedure space $i \in P$, $l \in P$

3.6.6. Equipment Alignment

The final constraint involves alignment of equipment to procedural requirements. As discussed in Section 3.4.10, scientists will have the opportunity to specify the specific procedural equipment and facilities infrastructure that they need in order to successfully conduct their research. This constraint is written in the following form:

$$R_{ji} \leq 1 + W_{ei} - N_{je}, \qquad 1 \leq e \leq E, 1 \leq j \leq J, 1 \leq i \leq P \tag{14}$$

This constraint compares the available equipment in each procedural space with the requested procedure equipment per job. Since these are Boolean integer parameters, there are four possible outcomes for this inequality. If the procedure space contains the required equipment (i.e. $W_{ei} = 1$), then the job can be assigned to the procedural space regardless of the need for this equipment. If a certain piece of equipment is not available in a procedural space (i.e. $W_{ei} = 0$), then the need for the equipment must be evaluated to determine compatibility between the requested job and the procedural space. When the equipment is not needed for the job, the procedural space can be assigned to the job without any issues. However, if the equipment is needed for a certain job (i.e. $N_{je} = 1$), then the procedural space will not be compatible with the job. In this case, R_{ji} must be constrained to a value of zero. The remaining three scenarios should not constrain the value of R_{ji}. The above constraint achieves this goal.

Equation 14- Variable Definition Review

Decision Variables:

R_{ji}	Procedural space assignment variable (1 if job $j \in J$ is assigned to procedural space $i \in P$)

Parameters:

N_{je}	Equipment to job alignment parameter (1 if equipment $e \in E$ is required to perform job $j \in J$)
W_{ei}	Procedure space to equipment alignment parameter (1 if equipment $e \in E$ is contained within procedural space $i \in P$)
e	Subscript used to denote a specific piece of equipment $e \in E$
j	Subscript used to denote a specific job $j \in J$
i	Subscript used to denote a specific procedure space $i \in P$
Ψ	A sufficiently large number

3.6.7. Constraint Equation Summary

Refer to Table 4 for a summary of the procedural space scheduling decision variables:

Table 4- Constraint Equations

Category	Equation	Range	Eqn #
Single Procedural Space Assignment	$\sum_{i=1}^{P} R_{ji} = 1, \quad 1 \le j \le J$	$1 \le j \le J$	*1*
Coupling Variables	$A_{jkil} \le R_{ji}$	$1 \le j,k \le J$ $1 \le i,l \le P$	*2*
	$A_{jkil} \le R_{kl}$		*3*
	$A_{jkil} \ge R_{ji} + R_{kl} - 1$		*4*
Time Bounds	$t_j \ge \gamma$	$1 \le j \le J$	*5*
	$t_j + \Delta_j \le \lambda$		*6*
Shared Room Restrictions	$(t_j + \Delta_j) - t_k + M_{jk}\Psi > 0$	$1 \le j,k \le J$	*7*
	$(t_j + \Delta_j) - t_k - (1 - M_{jk})\Psi \le 0$		*8*
	$M_{jk} + M_{kj} \ge A_{jkii}$	$1 \le j,k \le J$ $j \ne k$ $1 \le i \le P$	*9*
	$M_{jk} + M_{kj} \le 1$		*10*
Species Restrictions and Interactions	$R_{ji} \le \sum_{b=1}^{B} V_{jb} S_{bi}$	$1 \le i \le P$ $1 \le j \le J$	*11*
	$X_{il}\left(\sum_{b=1}^{B} \lvert V_{jb} - V_{kb} \rvert\right)(A_{jkil} - M_{jk} - M_{kj} + 1) \le \sum_{b=1}^{B} \lvert V_{jb} - V_{kb} \rvert$	$1 \le j,k \le J$ $j \ne k$ $1 \le i,l \le P$	*12*
Equipment Alignment	$R_{ji} \le 1 + W_{ei} - N_{je}$	$1 \le e \le E$ $1 \le j \le J$ $1 \le i \le P$	*14*

3.7. Objective Function

As discussed previously, scientists will specify the time that they prefer to begin their studies and the procedural space where they prefer to conduct this research. A best-case scheduling output would exactly match the requested start time and procedural space. A penalty will be incurred when this cannot be accommodated. The objective function is designed to capture the system wide penalty or "cost" incurred by scheduling decisions and has the form:

$$\text{Minimize: } \underbrace{(1-\alpha)\sum_{j=1}^{J}\sum_{i=1}^{P} c_j R_{ji}(1-Y_{ji})Z_{ji}(1-0.5\phi_j)}_{\text{Penalty due to room assignment}} + \underbrace{\alpha\sum_{j=1}^{J}|\omega_j - t_j|(0.5+0.5\phi_j)}_{\text{Penalty due to start time assignment}} \quad (15)$$

Subject to constraints (1) - (12) and (14)

Objective Function- Variable Definition Review

Decision Variables:

R_{ji}	Procedural space assignment variable (1 if job $j \in J$ is assigned to procedural space $i \in P$)
t_j	Assigned procedure start time for job $j \in J$

Parameters:

c_j	The number of cages associated with job $j \in J$
Y_{ji}	Procedure room preference parameter (1 if procedure room $i \in P$ is selected as the preferred location for job $j \in J$)
Z_{ji}	Logistical impact parameter (A positive integer which represents the logistical impact between the holding room containing animals for job $j \in J$ and procedure room $i \in P$)
α	Objective function weighting factor
ϕ_j	Research start time preference indicator for job $j \in J$
ω_j	The preferred start time for job $j \in J$ as defined by the scientist
j	Subscript used to denote a specific job $j \in J$
i	Subscript used to denote a specific procedure space $i \in P$

Minimizing the objective function will provide the optimal research schedule that vivarium managers can use to efficiently allocate resources within their facilities. One important feature of the objective function is the ability for vivarium managers to adjust the penalty assignment weightings to reflect their unique research needs and facility requirements. This is accomplished

using the α parameter. In addition to prioritizing one penalty over the other, adjusting the α parameter provides the ability to equalize these two quantities since different units are used to calculate the penalty values (i.e. distance versus time).

Looking at the objective function in more detail, the room assignment penalty calculation will evaluate all combinations of job assignments and procedural space pairings. For each instance, the function will first evaluate if a job pairing exists. When a job is not assigned to the procedural space under consideration, R_{ji} will equal zero and no penalty will be assigned. However, when a job has been assigned to a procedural space, R_{ji} will equal one. This will permit the penalty associated with the room assignment to be evaluated. Next, the procedural space preference parameter Y_{ji} is evaluated for each unique procedure space and job combination. Since this parameter equals one when a procedural space has been selected as preferred, a minimization strategy will be achieved when Y_{ji} is subtracted from one. If the algorithm is evaluating a room assignment that differs from that specified by the scientist, the quantity (1- Y_{ji}) will reduce to one, creating a scenario where a penalty cost will be attributed to the assignment. Next, the Z_{ji} parameter will determine the magnitude of the penalization assuming R_{ji} is equal to one and Y_{ji} is equal to zero. The magnitude of penalty assigned will be based on spatial units or spatial unit equivalents since Z_{ji} measures the distance between the holding room of interest and the procedural space being evaluated. This value will be multiplied by the number of cages associated with the job (c_j). Impact of the research start time preference indicator ϕ_j will be discussed in a later section.

Moving to the start time assignment penalty calculation, this portion of the objective function will evaluate all job assignments to determine the magnitude of mismatch between the requested start time ω_j and the assigned start time t_j. Of interest is the difference between these two values. This analysis assumes that an assigned start time that occurs earlier than the requested start time carries the same penalty as an assigned start time that occurs after the requested start time. Since the scheduling model allows both instances to occur, the value of (ω_j - t_j) can be either negative or positive. To ensure that the penalties are additive, the objective function uses the absolute value of the difference to account for positive and negative values of the difference expression.

The research start time preference indicator ϕ_j provides additional weighting within the objective function to account for differences in research priorities. As discussed in Section 3.4.12, a scientist will specify their preference for either beginning their research closer to the requested start time or conducting their research in their preferred procedural space(s). The parameter ϕ_j will amplify either the room assignment penalty or the start time penalty portion of the objective function based on input from the scientist submitting the work request. The objective function has been developed based on ϕ_j equaling one when start time is prioritized. Since this is a binary parameter, two scenarios are possible for objective function weighting, as shown in Table 5.

Table 5- Effect of research preference on objective weighting

ϕ_j Value	Penalty Function	Weighting Expression	Weighting Value
$\phi_j = 1$ **(Start time preference)**	Start Time	$(0.5 + 0.5\phi_j)$	1.0
	Room Assignment	$(1 - 0.5\phi_j)$	0.5
$\phi_j = 0$ **(Research space preference)**	Start Time	$(0.5 + 0.5\phi_j)$	0.5
	Room Assignment	$(1 - 0.5\phi_j)$	1.0

As Table 5 indicates, the weighting has been developed to double the impact that the weighted penalty contributes to the objective function versus the non-weighted penalty. This weighting strategy was arbitrarily selected and can be customized to meet specific operational requirements.

3.8. Chapter Summary

This chapter focused on the development of a complex optimization model. This model was prefaced with an overview of environmental factors that influence research scheduling at NIBR. A literature review was provided to study applicable optimization research that has been conducted in other industries. This review identified the healthcare and airline industries as having scheduling challenges with attributes similar to those in the pharmaceutical industry. Of particular use are optimization models that focus on the airport gate assignment problem (AGAP). Research from two groups was leveraged to develop an optimization model that accommodates unique operational challenges found in the pharmaceutical industry. Parameters and decision variables specific to the in vivo research room scheduling problem (RAP) were presented. These parameters and variables were used to develop constraints and an objective function. Techniques have been used that transform the quadratic assignment problem into a mixed integer programming model with linear constraints and a linear objective function. A multi-criteria objective function uses the researcher's preference to optimize both room assignments and procedure start time. The following chapter will discuss an alternative algorithm for executing this optimization model.

This page has been intentionally left blank

4. Algorithm Development

4.1. Chapter Overview and Algorithm Approach

The optimization approach developed in the previous chapter provides a method for identifying a solution that guarantees minimization of the objective function. AGAP research (Lim et al. (2005), Ding et al. (2004), Xu and Bailey (2001)) indicates that computational solving tools used to find exact solutions sometimes have trouble reaching an optimal solution within a desired timeframe due to the structure of the problem and the number of inequalities that must be evaluated. These concerns necessitate the development of an alternative approach to optimizing the RAP. As will be discussed in the next chapter, the ability of a traditional solver packages (e.g. CPLEX) to optimize the RAP should be compared against an alternative method in order to determine the best implementation strategy. The alternative approach recommended to optimize the RAP uses a tabu search meta-heuristic to find a near-optimum solution through iterative neighborhood searches. This chapter discusses implementing a tabu search algorithm, including the move strategies and subroutines required to execute this algorithm. In addition, a heuristic approach is proposed for developing an initial feasible solution, which is used as input into the tabu search algorithm. The same constraints developed in the previous chapter are incorporated into this scheduling algorithm.

4.2. Initial Schedule Development

Developing an initial feasible solution to the RAP is more difficult compared to the AGAP since there are several key constraints that are not addressed using a simple greedy algorithm to align room schedules. In addition to timing overlap, room assignments must also consider species and equipment related constraints. The AGAP model developed by Ding et al. (2004) assigns flights to gates based purely on the planned departure and arrival times of each flight. This approach cannot be used since jobs can typically only be performed in a subset of available rooms. To address this complexity, an algorithm has been developed that generates an initial solution taking into account room assignments to ensure jobs are being placed into rooms that are compatible with the requests. The algorithm assumes that rooms selected as "preferred" by the scientists contain the equipment required for their procedures. In addition, this algorithm assumes that the preferred rooms are compatible with the species associated with each job. Although these assumptions address equipment and species alignment with each procedural space, they do not ensure that different species will not be assigned to adjacent procedural spaces at overlapping times. In order to prevent this from occurring, a subroutine called *RoomCheck(j, l, τ)* is employed.

RoomCheck(j, l, τ) evaluates the compatibility of job *j* with placement in procedural space *i* and a starting time of τ. The subroutine is set to an initial value of "true". In all cases where the

preferred procedural space is within a shared procedural room ($X_{il} = 1$), the subroutine performs two evaluations. First, overlap between job j and each job k in the adjacent procedural space l is evaluated. If overlap does not exist for all jobs, *RoomCheck(j, l, τ)* returns a value of true. If an overlap exists, the subroutine compares the species associated with job j and the overlapping job k. A value of "true" is returned if the species for these two jobs are identical. Otherwise, the value is changed to "false", which indicates that different species would potentially be in the same area at overlapping times.

The algorithm used to determine the initial feasible solution is a more complex version of the greedy algorithm developed by Ding et al. (2004). The algorithm begins by organizing all requested jobs for a given day in ascending order according to requested start time ω_j ($1 \leq j \leq J$). The variable d_i represents the earliest time that procedural space i is available. To begin the algorithm, d_i is initialized to the lower scheduling bound ($d_i = \gamma$) for all procedural spaces i. This ensures that jobs are not scheduled outside standard operating hours. The first job is automatically assigned to the preferred procedural space. In cases where more than one procedural space is marked by the scientist as preferred, the spatial relationships between the holding room and procedural spaces are used to determine the assignment. Specifically, the first job is assigned to the preferred procedural space with the lowest Z_{ji} value. The assigned start time for this job now equals the requested start time ($t_j = \omega_j$). Once this job is assigned to a procedural space, d_i is updated to equal the completion time of this job ($d_i = t_j + \Delta_j$). Next, the *RoomCheck(j, l, τ)* subroutine is used to evaluate the set of preferred procedural spaces for the second job. For each preferred procedural space, the algorithm first determines the best available start time (τ). As discussed previously, the X_{il} parameter will be used to determine which procedural spaces are in close proximity to procedural space i. Using this data, *RoomCheck(j, l, τ)* will return either a true or false value. The subset of procedural spaces designated as true will then be evaluated based on the preferred start time. As with the first job, the Z_{ji} parameter will be used to determine room assignments in cases where the assigned start time exactly matches the requested start time. In the event that a conflict exists for all identified procedural spaces ($d_i > \omega_j$), the procedural space with the smallest start time offset will be assigned to this job. In this scenario, the Z_{ji} parameter is not be used to determine room assignments. When the requested start time cannot be accommodated, the assigned start time will equal the room availability time ($t_j = d_i$). This assignment process is repeated for all requested jobs.

The algorithm discussed above takes into account the start time and duration of each job. In addition, the algorithm compares the projected completion time to the upper scheduling bound. The procedural space is removed from consideration if the completion time for the job is greater than the upper scheduling bound ($t_j + \Delta_j > \lambda$). Jobs will need to be assigned to a waitlist when research requests outstrip the procedural space capacity for the entire facility or for a procedural space containing unique equipment (e.g. MRI). This waitlist is analogous to assigning flights to the apron when a gate is not available. Minimizing the number of jobs assigned to the waitlist is critical since delaying research impacts perceived customer service and research output. The

waitlist can be modeled using several different approaches. One simple method is to have rooms designated as waitlist placeholders. To ensure the optimization model will function correctly, these waitlist procedural spaces will have extremely high Z_{ji} parameter values, which will penalize the objective function when a job is assigned to one of these rooms. When no preferred procedural space has sufficient capacity to schedule a job, the job will be moved to the waitlist. The tabu search algorithm discussed in the next section will attempt to reduce the number of jobs assigned to the waitlist.

4.3. Neighborhood Search Moves

Once the initial feasible solution has been developed, a tabu search algorithm is used to optimize the schedule. This algorithm uses several neighborhood moves to develop a near optimum schedule. These moves and related subroutines are discussed below.

4.3.1. Subroutines

The neighborhood moves described later in this section require eighteen subroutines in order to properly execute their desired functions. Several of these subroutines are based on models developed by Lim et al. (2005). A comparison of the subroutines found in this paper versus the subroutines developed by Lim et al. will yield several important differences. First, Lim et al. assume that flight windows exist, which are larger in duration compared to the actual flight turnaround times. In addition, their model assumes that shifting a flight to the front portion (left most) of this flight window provides the optimum scheduling solution for the specific flight. Scheduling rooms differs from this application in that there are not specified windows that the research must be completed within. Although the service level is directly impacted when the assigned start time differs from the requested start time, a bounded window does not exist. In addition, the impact to customer service is symmetrical for deviations from the requested start time. In other words, the magnitude of service level degradation is equal for one job where the assigned start time is one hour prior to the requested start timè versus another job where the assigned start time is one hour after the requested start time. The unique structure of the RAP is reflected in these subroutines, which are summarized in Table 6. Detailed information regarding these subroutines can be found in Appendix A.

Table 6- Subroutine summary

Subroutine	*Description*
RoomCheck(j, l, τ)	Evaluates the compatibility of job *j* with placement in procedural space *i* and a starting time of *τ*.
CompatCheck(j, l)	Determines if the species for job *j* is compatible with the species restrictions for procedural space *l*.
ShiftRight(j, i, τ)	Shifts the start time of job *j* assigned to procedural space *i*. The magnitude of this rightward shift shall be *τ* time units.
AttemptShiftRight(j, i)	Determines the latest time that job *j* can begin in procedural space *i*.
ShiftLeft(j, i, τ)	Shifts the start time of job *j* assigned to procedural space *i*. The magnitude of this leftward shift shall be *τ* time units.
AttemptShiftLeft(j, i)	Determines the earliest time that job *j* can begin in procedural space *i*.
GapMeasure(j, i)	Determines the difference between the assigned start time and requested start time for job *j* in procedural space *i*.
OptimizeRight(j, i)	Attempts to shift the assigned starting times beginning at job *j* in procedural space *i* to the right (later).
OptimizeLeft(j, i)	Attempts to shift the assigned starting times beginning at job *j* in procedural space *i* to the left (earlier).
EndPrev(j)	Determines the ending time (i.e. start time plus the job duration) for the job that is scheduled just prior to (earlier) job *j*.
NextStart(j)	Determines the starting time for the job that is scheduled just after job *j*.
PrevJob(j)	Determines the job that precedes job *j*
NextJob(j)	Determines the job that follows job *j*
InitializeInterval(X, j)	Creates an interval that is defined by the start and end times associated with job *j*. The variable *X* represents the name of this interval.
ExtendRight(X)	Increases the size of interval *X* by changing the end point of the interval to a later time. This new time point is the ending time of the following job.
AttemptExtendRight(X)	Returns a Boolean value that indicates the ability to perform an *ExtendRight(X)* operation for interval *X*.
ExtendLeft(X)	Increases the size of interval *X* by changing the end point of the interval to an earlier time. This new time point is the starting time of the preceding job.
AttemptExtendLeft(X)	Returns a Boolean value that indicates the ability to perform an *ExtendLeft(X)* operation for interval *X*.

4.3.2. The Insert Move

The Insert Move is denoted as *insert(j, i)* $\rightarrow$ *(j, l)* and is used to move a single job *j* from its current assigned procedural space *i*, to a new procedural space *l* ($i \neq l$). For this evaluation, the job *j* and the new procedural space *l* are both randomly selected. The current assigned procedural space *i* can be obtained once job *j* is known. The insert move was originally proposed by Xu and Bailey (2001) and later adopted by Ding et al (2004). Both implementations used a static approach where the time duration of the inserted flight is compared against flights that are already scheduled for a particular gate. With the Xu and Bailey algorithm, the flight is allowed to be inserted only if there is no overlap with existing flights. Lim et al. (2005) have developed a more complex insert move algorithm that allows both the inserted and existing flights to shift within a flight window. This enables a dynamic scheduling strategy and allows flights to be accommodated in schedules that would have previously prevented the move from occurring.

Two key differences exist between the scheduling model developed by Lim et al. and allocation of procedural space within a research lab. First, in the Lim et al. model, preference is given to flights that are assigned as close as possible to the front (left-side) of their flight window. Intuitively this scheduling strategy makes sense since it provides a high likelihood for there being a gate available when the flight arrives. However, defined time windows do not exist in the RAP model. Although jobs can be shifted from the requested start time, there are no bounds which prevent the scheduling of these jobs. In addition, the preferred assigned start time for a given job aligns with the requested start time. As the assigned start time begins to deviate from the preferred start time, the difference between these two times determines the magnitude of impact to customer service, not the direction of the difference (i.e. earlier vs. later). The second difference is the complex constrains that exist for the RAP. Not only do start times and durations need to align for this problem, other constrains such as species interactions and equipment alignment need to be taken into consideration. For these reasons, the insert move algorithm for the RAP model is more complex compared to the AGAP.

The insert move algorithm developed for the RAP is divided into five primary steps, as shown in Figure 17. First, job *j* and targeted procedural space *l* are randomly selected from the set of all jobs and procedural spaces. After the key variables are defined, the insert move algorithm utilizes the *CompatCheck(j, l)* subroutine to determine if the job to be inserted is compatible with the targeted procedural space in terms of both species and equipment. If compatibility is verified, the algorithm then determines the best location for inserting job *j* into the schedule for procedural space *l*. This is accomplished by comparing the job transition time to the requested start time for job *j*. The transition location that has the closest proximity to the requested start time for job *j* will be selected as the insertion location. The median time between two jobs (t_m) is used as a reference of this analysis.

Once the insertion location is selected, the algorithm compares the job transition duration to the duration of job *j*. A simple insertion process can be executed if the duration of job *j* is less than

the time interval between the two previously scheduled jobs. This insertion process involves placing job j directly after the preceding (earlier) job. Prior to making the insertion, the subroutine *RoomCheck(j, l, τ)* is used to determine if there are compatibility issues with other jobs that are located in the same procedural room. If no conflicts exist, job j is removed from the schedule for procedural space i and inserted into the schedule for procedural space l. To complete the insertion process, the *OptimizeRight(j, i)* subroutine is used to position the newly inserted job at an optimum location within the available space.

Inserting job j into a schedule where the time interval between the two existing jobs is smaller than the duration of job j is more challenging compared to the latter scenario. In this constrained case, the overall flexibility of the neighborhood will be evaluated to determine if an opportunity exists to fit job j into the schedule for procedural space l. To begin this analysis, the *AttemptShiftLeft(j, i)* and *AttemptShiftRight(j, i)* subroutines are used to calculate the amount of additional space that can potentially be added to the existing interval. If the additional space provided by these potential moves exceed the duration of job j, both jobs that bound the interval are shifted in opposite directions using *ShiftRight(j, i, τ)* and *ShiftLeft(j, i, τ)*. As shown in Step 2 of Figure 17, this results in a larger interval. As with the previous case, job j is inserted into this space (Step 3), after which the placement is optimized (Step 4). In addition, the placement of the job situated to the right of job j is optimized using the *OptimizeLeft(j, i)* subroutine.

The final phase of the insert move algorithm involves optimizing procedural space i, from which job j was removed (Step 5). This is accomplished in a manner similar to the above process. First, the *OptimizeLeft(j, i)* subroutine is used to shift the job that followed job j. Next, the *OptimizeRight(j, i)* subroutine is used to shift the job that proceeded job j in this procedural space. These two functions will potentially reduce the interval created when job j was moved. Schedule constraints and job alignment will prevent many insertions from occurring. For this reason, the insert move function will iterate until a feasible solution is found. Once a successful move is executed, the objective function can easily be updated to determine the performance of this move. Appendix B describes in detail the Insert Move process.

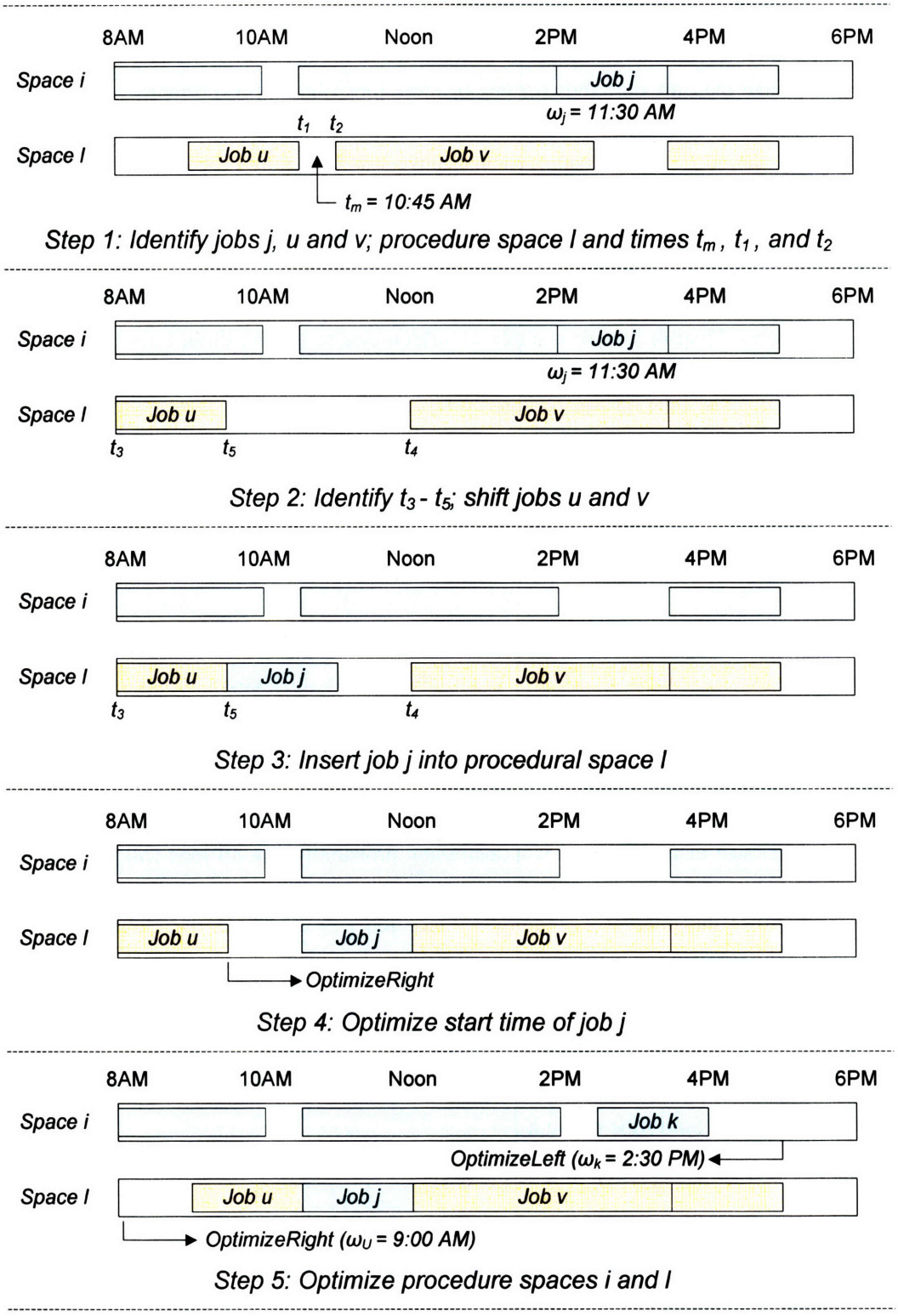


Figure 17- Insert Move algorithm illustration

4.3.3. The Interval Exchange Move

The Interval Exchange Move is denoted as *IntervalExchange(A, i)* $\leftrightarrow$ *(B, l)* and is used to exchange the time durations for one or more jobs (interval *A*) currently scheduled in procedural space *i* with one or more jobs (interval *B*) currently scheduled in procedural space *l* ($i \neq l$). An early version of the interval exchange move was originally proposed by Xu and Bailey (2001). The two algorithms employed by Xu and Bailey (*Exchange I Move* and *Exchange II Move*) are fairly rigid since the interval bounds are not flexible and the move must be between two single flights or two pairs of flights. Ding et al. (2004) improved upon this approach by developing the *Interval Exchange Move* that defined the specific intervals based on compatibility between the intervals. Ding et al. use a series of subroutines to extend these intervals in order to achieve compatibility. One drawback of this method is the number of flights that are selected for the interval. The interval can potentially be defined as a larger number of flights since the initial selection process is random. In addition, the size increases with each iteration of the algorithm assessment. Lim et al. (2005) also use an *Interval Exchange Move* function, however their implementation does not achieve compatibility by changing the number of flights included in the interval. Instead, Lim et al. use shift functions to move the interval within the predefined flight windows.

The interval exchange processes used for the RAP is most closely related to the algorithm developed by Ding et al. (2004). One of the primary differences between these two algorithms is the approach used to define the initial interval size. Where Ding et al. select intervals by randomly choosing two pairs of same-gate flights, the method employed for this analysis begins by randomly selecting one job *j* that is associated with procedural space *i*. This approach is used since many jobs will be constrained by procedural space and species requirements. As the interval size increases, there is a decreasing probability that all jobs will be compatible with the targeted procedural space. This approach begins with the smallest interval size possible (i.e. 1:1) and grows the interval as required. Once job *j* is identified, the algorithm randomly selects procedural space *l*, where $i \neq l$. The algorithm then identifies the first job *k* in procedural space *l* that has an interval overlap with job *j*. Next, four time points are defined for each interval. These time points correspond to the following features:

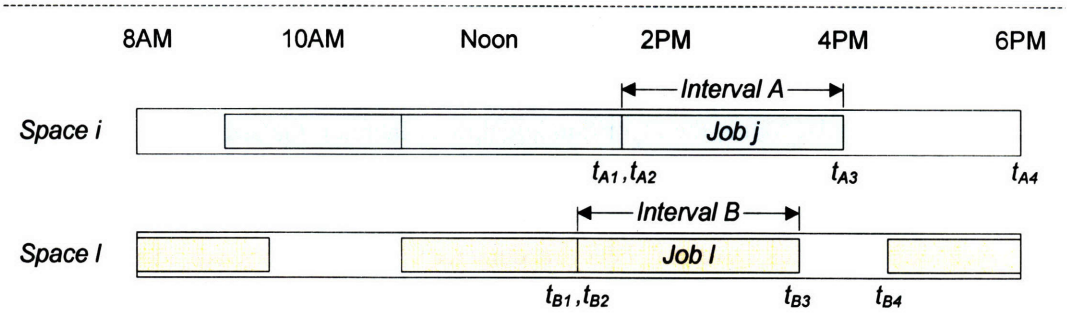
T_{X1}: Completion time of the job that precedes job *X*

T_{X2}: Start time of job *X*

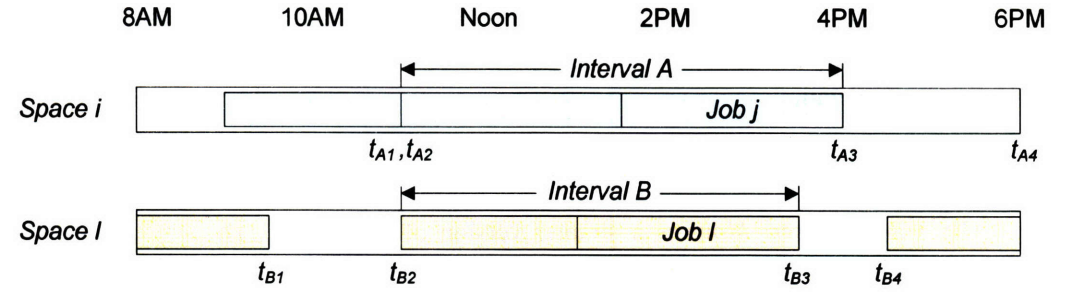
T_{X3}: Completion time of job *X*

T_{X4}: Start time of the job that immediately follows job *X*

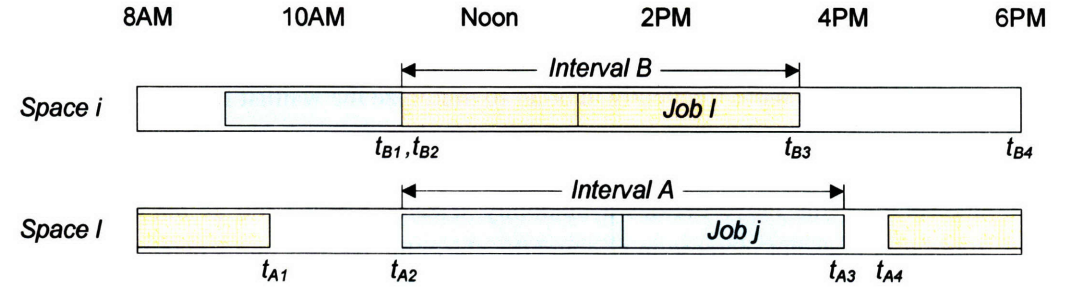
where *X* is a generic job reference that will change as the size of the interval increases. Once the initial interval is known, the algorithm uses the *RoomCheck(j, l, τ)* subroutine to verify the compatibility of these rooms with the targeted procedural space. Incompatibility results in the selection of a new interval. Step 1 in Figure 18 provides a visual representation of these intervals and time points.



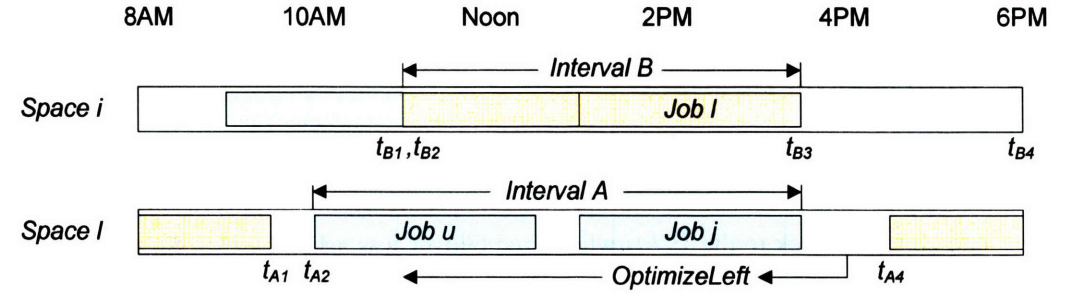
Step 1: Identify intervals A and B; determine interval time points



Step 2: Determine the smallest compatible intervals for A and B



Step 3: Exchange intervals A and B



Step 4: Optimize intervals A and B (assume ω_u = 10:00 AM and ω_j = 1:00 PM)

Figure 18- Interval Exchange Move algorithm illustration

If compatibility is verified, the algorithm attempts to align the intervals by ensuring no conflicts exist with jobs that are outside of the interval. This is accomplished by verifying the following four inequalities are met: $t_{12} \geq t_{21}$, $t_{22} \geq t_{11}$, $t_{14} \geq t_{23}$, $t_{24} \geq t_{13}$. If any one of these four inequalities results in a false value, the algorithm attempts to increase the size of the interval to include the conflicting job (Step 2). This is accomplished by verifying compatibility with the new procedural space, as well as verifying interval size extendibility (i.e. ensuring that the upper or lower scheduling bounds have not been encountered). If no issues are identified, the interval is extended and an additional check is completed to verify that all conflicts have been eliminated. This process is iterated until compatible intervals are developed or incompatibility is verified. If alignment between the intervals is achieved, the intervals are swapped (Step 3) and the *OptimizeRight(j, i)* or *OptimizeLeft(j, i)* subroutines are used to adjust the position of these jobs towards their preferred start times (Step 4). Appendix C describes the Interval Exchange Move process in detail.

4.3.4. The Waitlist Insert and Exchange Moves

The *Waitlist Insert Move* is a simpler version of the *insert(j, i)* $\rightarrow$ *(j, l)* algorithm described previously. This function identifies a job that is currently on the waitlist and attempts to find a procedural space schedule for which this job can be inserted into. The same base algorithm is used with the only difference being the final optimization process. This final optimization is not required since the waitlist does not have a schedule that can be optimized. Defining the *Waitlist Insert Move* as a stand-alone function is important since it allows the tabu search algorithm to utilize this move more frequently in order to minimize the waitlist length.

The *Waitlist Exchange Move* is a simplified version of the *IntervalExchange(A, i)* $\rightarrow$ *(B, l)* algorithm. This move defines a randomly selected waitlist job as *Interval A*. Once defined, *Interval A* does not increase in size and the algorithm focuses on increasing the size of *Interval B* to accommodate the waitlisted job. If compatibility is achieved, the waitlisted job will be inserted into the procedural space schedule and the job(s) contained within *Interval B* will be moved to the waitlist.

4.4. Tabu Search Heuristic

Although tabu search (TS) has only gained popularity in recent years as a viable method for solving complex scheduling and optimization problems, the origins of this algorithmic technique can be traced back to the 1970s when Fred Glover was asked by U.S. Strategic Air Command to develop a strategy for responding to a hypothetic nuclear strike. Computational methods at the time could not provide an accurate solution in the required timeframe (5 minutes). In the face of very stiff skepticism, Glover developed an algorithm that was able to find an effective retaliatory response in less than 30 seconds. Glover recounts this story in his recent article *Tabu Search-Uncharted Domains* (2007). Glover's method utilized adaptive memory or the ability for the algorithm to remember changes that had been made in the past in order to make more accurate

decisions. The use of adaptive memory became a hallmark for this category of optimization techniques. The name "Tabu Search" was coined in 1986, after which this technique quickly gained popularity among scientists and engineers attempting to optimize a wide variety of systems. Glover attributes this growth in popularity to the evangelizing of TS techniques by a core group of researchers and less to the new nomenclature used to describe this optimization method.

A TS approach has been selected to address the RAP based on the successful application of TS algorithms for solving other combinational optimization problems, including the AGAP. Xu and Bailey (2001) developed one of the first AGAP scheduling algorithms to utilize a TS methodology. Their TS scheduling method produced a 24.7% average savings advantage versus static assignment methods. Ding et al. (2004) and Lim et al. (2005) also utilized TS methods for addressing their more complex versions of the AGAP. After comparing a TS scheduling methodology with other scheduling methods including memetic algorithms and genetic algorithms, Lim et al. (2005) concluded that, "TS is a suitable approach to tackle the AGAP". Their results showed better performance for the TS based scheduling algorithm versus the other approaches, for both large and small test instances.

A tabu search algorithm works in an iterative fashion to make small changes to a solution and measure the effectiveness of the new solution compared to the solution from which it was derived. The word "tabu" is used to describe this optimization method since the adaptive memory acts to prevent the algorithm from selecting neighborhood solutions that might lead to neighborhoods that have already been explored. The adaptive memory in a tabu search algorithm takes the form of a tabu list that contains the inverse moves for neighborhood moves that have been made in the past. Any solution on the tabu list cannot be made unless the objective function value obtained using the new solution exceeds an *aspiration criteria*. Many times the aspiration criteria is equal to the best objective function value for all measured solutions. The length of the tabu list is finite and is determined by a parameter called the *tabu tenure*. The tabu tenure determines the number of iterations for which a neighborhood move is considered tabu. As a solution evolves through iterative neighborhood changes, the metamorphosis will be sufficiently great to allow the tabu listed move to be removed from the list without worry that a past neighborhood will be re-evaluated. When different move subroutines are available for finding neighborhood solutions, the algorithm can randomly select a move strategy or control the application of these subroutines. *Intensification* and *diversification* strategies can be employed when the subroutines are used in a controlled fashion. Intensification occurs when a neighborhood is explored in greater depth using moves that change only small aspects of the neighborhood. Intensification should be used when a solution is thought to be near-optimal. In contrast, diversification is used to develop a neighborhood solution that differs significantly from its predecessor. Diversification is helpful for moving away from local optima.

The focus of this chapter is to provide a framework for developing a scheduling algorithm that can successfully be used to schedule in vivo research activities. Developing and testing an actual algorithm and web-based scheduling tool are beyond the scope of this discussion. However, the framework will be discussed in a context that assumes a computerized tool and user interface exist for collecting and organizing research data. To begin the tabu search, research requests for a 24-hour period are entered into the RAP scheduling tool. Next, search parameters are entered by the user, who will typically be the vivarium manager or other member of the operations team responsible for coordinating assignment of resources within the vivariums. These parameters include the following values:

- Tabu Tenure (τ): The number of inverse moves stored in the adaptive memory (i.e. the tabu list).
- Neighborhood Solutions (N): The number of potential solutions evaluated for each iteration of the TS algorithm.
- "No Change" Iteration Limit ($MAXNC$): The number of iterations that can occur without an improvement in the objective function value before the algorithm terminates and declares the final solution.
- Maximum Iteration Limit ($MAXINT$): The maximum allowable number of total iterations that can occur before the algorithm terminates and declares the final solution.

When the tabu search begins, the iteration counters are both initialized ($INT = 0$ and $NC = 1$). Next, an initial feasible solution is generated using the process discussed in Section 4.2. The accuracy of this initial solution is critical since subsequent moves are predicated on the assumption that this solution conforms to all applicable scheduling constraints. The objective function value is calculated using this initial solution and then stored as the maximized or minimize objective value (C_{BEST}). The variable C_{BEST} will be updated throughout the search as better solutions are identified.

Once the search algorithm has been initialized, the algorithm selects the first move strategy that will be used to develop a list of neighborhood moves. For the RAP, the tool randomly selects from the Insert Move, Interval Exchange Move, Waitlist Insert Move and Waitlist Exchange Move strategies. Since these move strategies are randomly selected, the algorithm does not take advantage of intensification and diversification techniques. Using the selected move strategy, the first N neighborhood solutions are identified and evaluated. The solution that corresponds to the lowest objective function value is compared against the initial solution. If this solution has a lower objective function value, C_{BEST} is updated to equal this value, the solution value ($X_{INT, N}$) is set equal to X_{BEST} and NC is set equal to one. Otherwise, C_{BEST} and X_{BEST} remain unchanged, and both iteration counters are updated. The inverse move is then entered into the tabu list.

The process is then repeated, with the selection of a new move strategy and neighborhood solutions. The process becomes more complex compared to the first iteration since the solution

with the lowest objective function value must be compared to the tabu list. If the move is not included in the tabu list, the same process can be followed where $X_{INT,\ N}$ is compared to C_{BEST}. However, if the move is included in the tabu list, the solution must be rejected if it does not exceed the aspiration criteria ($C_{INT,\ N} < C_{BEST}$). If the solution is rejected, the next lowest solution must be selected and evaluated. Once an acceptable solution is found, the objective function value for $X_{INT,\ N}$ is compared to C_{BEST}. C_{BEST} and X_{BEST} are updated if a better solution is identified. The algorithm will then determine if one of the two termination criterions are met. For instances where the solution was not improved, the algorithm compares the *NC* counter to the *MAXNC* value to determine if a sufficient number of iterations have occurred without improvement in the objective function. If this is the case, the algorithm will terminate. In addition, the algorithm will also terminate if the *INT* counter surpasses the *MAXINT* value. If termination does not occur, the algorithm updates all relevant counters and variables. At the end of each iteration, the tabu list is update to include the inverse move that was selected during the current iteration and remove the outdated move. Figure 19 provides a detailed flow diagram of the algorithm logic.

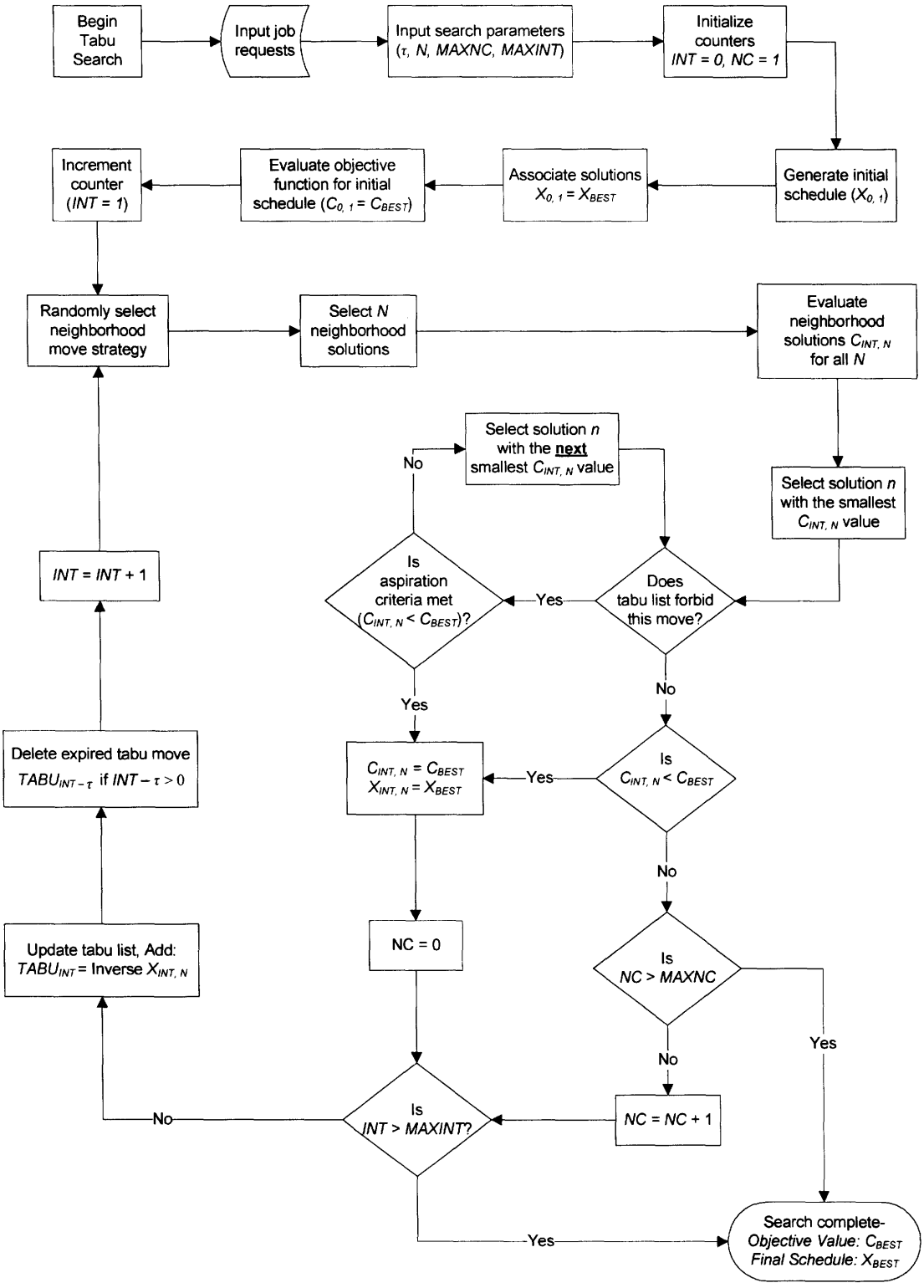


Figure 19- Tabu Search Logic Flow Diagram

Optimization of the algorithm parameters should be performed prior to production implementation. Values such as the Tabu Tenure (τ), Neighborhood Solutions (N), and iteration limits (*MAXNC* and *MAXINT*) should be varied using a design of experiments analysis to determine the optimal values to use for this model. As a reference point, Lim et al. (2005) found that the following values performed well for their tabu search algorithm:

- Tabu Tenure (τ): 10
- Neighborhood Solutions (N): 100
- "No Change" Iteration Limit (*MAXNC*): 10,000
- Maximum Iteration Limit (*MAXINT*): 1,000,000

4.5. Chapter Summary

This chapter has focused on the development of a heuristic based algorithm that is used to identify a near-optimum solution for the RAP. A tabu search algorithm is proposed, which leverages four move strategies and an iterative search technique. The Insert Move, Interval Exchange Move, Waitlist Insert Move and Waitlist Exchange Move strategies are search techniques rooted in algorithms developed for the AGAP, but significantly modified to address the unique challenges associated with scheduling in vivo research activities. A greedy heuristic is utilized to develop an initial feasible solution for input into the tabu search process. A series of subroutines are proposed which are used to execute the greedy algorithm and the four move strategies. These subroutines ensure job compatibility, shift jobs within assigned procedural space, group jobs within specific intervals and optimize placement of jobs.

This page has been intentionally left blank

5. Recommendations and Conclusions

This chapter discusses an implementation strategy for integrating the concepts discussed in this paper, as well as deploying a functioning scheduling tool that can be utilized by the research community at NIBR. Although several important concepts and models are explained in this thesis, additional research and model integration must occur in order to develop and deploy a best in class solution. I highly recommend that these activities focus on two primary areas:

- ***Joint Optimization Model Development:*** In addition to optimizing procedural space assignment, assignment of holding room space can also benefit from an optimization algorithm. Combining optimization of holding room assignments and procedural room assignments will provide a powerful tool that will greatly enhance vivarium operations.
- ***Stochastic Procedure Duration Modeling:*** A deterministic approach has been used to develop the model discussed in Chapter 3. Scientists input the time required to complete their research (Δ_j), which is used to determine the procedure time interval. No estimate for Δ_j is perfect and the actual duration for a given type of procedure will fall within a range of values from which a distribution can be created. Based on this distribution, the variability in procedure duration can be used to turn the deterministic model into a stochastic model, which will more accurately assign procedure start times.

Aspects of these activities are discussed in the next two sections.

5.1. Recommendations

With the development of a useable scheduling model, NIBR must determine how to develop and integrate a fully functional web-based tool into in vivo research operations. This requires tight collaboration with Information Technology (IT), disease area representatives, and third party programmers. Four steps are recommended to complete this process:

Step 1: Prototype Development

Building a coalition of supporters is critical for the success of this project. A handful of scientists from various disease areas have already been engaged on this project and provided useful feedback, which was very positive. These individuals see the value that this process provides and understand how their research will directly benefit from using a centralized resource management system. These scientists stressed the need for this system to be flexible and efficient. They indicated that redundant data entry would not be tolerated by the research community. Moving forward, additional feedback from a larger audience will provide the commitment required to move this project forward. As the project gains momentum, a cross-functional team consisting of disease area representatives should be convened in order to begin this feedback process.

Since the algorithms developed in this paper have not been validated, the IT and programming team should focus their efforts on creating a no-frills version of the tool to test the overall algorithmic efficiency. As discussed in the previous chapter, a design of experiments approach should be utilized in order to determine the best values for τ, N, *MAXNC*, and *MAXINT*. Simulations should be used to test various facility configurations. In addition, the tabu search algorithm should be directly compared against a CPLEX solution in order to determine differences in objective function values and processing times. The data generated from this comparison, as well as implementation and sustaining costs should be used to select the most attractive method (CPLEX solver versus tabu search) for optimizing the RAP.

Additional research should be pursed in parallel with these activities that focuses on developing a probabilistic model to address fluctuations in procedure durations. The current deterministic model requires scientists to estimate the duration for the procedure they plan to perform (Δ_j). This can produce two undesirable outcomes. First, the researcher might "pad" their estimates to ensure enough time is allotted for their procedure. This creates inefficiency by allowing the research team to become complacent in execution of their work. It also increases the likelihood of having procedure space idle during peak hours of operation. In addition, complications sometimes arise, which can prolong the duration of a procedure. This will create a scheduling conflict if another procedure is assigned to the same procedure space directly after the estimated completion time of the first procedure. Since the timing of procedures can impact the results, it is important to minimize the occurrence of these conflicts. Developing a probabilistic model that uses past data to determine the appropriate procedure durations will help minimize the impact that procedure fluctuations have on research scheduling and coordination.

Step 2: Limited Pilot

Adoption of this scheduling and coordination tool is dependent on a positive user experience. The ability to successfully launch this tool could be impacted if significant bugs exist that provide excuses for the research community to continue using their existing processes. In order to minimize the risk of this occurring, the tool should undergo extensive user testing prior to launch. In addition, the tool should not be launched to the entire organization at one time. Instead, a phased deployment strategy should be used. This will allow for more personalized engagement during the launch process. In addition, issues with the software can quickly be addressed when dealing with a subset of users. During this phase, the software will need to constrain job scheduling to specific research areas. This is required since the algorithm will not function properly if it cannot control allocation of all resources.

Step 3: Full Adoption and Centralization

A phased implementation strategy will eventually lead to full process adoption. After any transient software issues are addressed, the physical integration of research activities and operations should quickly begin. This involves combining like assets (i.e. animals and equipment) into common spaces to take advantage of risk pooling. The centralization and

integration strategy will depend on the facility layout and the quantity of animals that each group plans to use in the future. As these processes transition, care must be taken to prevent work-arounds. Work-arounds occur anytime the process is circumvented. LAS must not encourage or support work-around behavior. If LAS is vigilant, the research community will quickly learn that following standard procedures produces quality results with minimal investment.

Step 4: Integrate into NIBR Processes

Many activities in the R&D process have both upstream and downstream dependencies. The scheduling and coordination process discussed in this paper addresses only the dependencies that are contained within in vivo research operations. The next logical step is to provide a seamless management system capable of managing these external relationships. For example, a research project might be dependent on the development of a compound that is scheduled to be completed by the formulation group. Understanding this dependency, scheduling the tasks appropriately and proactively adjusting research schedules for any delays would benefit the resource allocation process. Although the benefits of coordination are enormous, integrating these workflows will require a significant amount of coordination with many functional groups across NIBR. Currently a team in Emeryville is developing a workflow tool internal to NIBR called *Animal Workflow*. This software is very complimentary to the scheduling tool discussed in this paper and integration of these two platforms should be pursued.

5.2. Future Research

After researching this topic, it became evident that there is a huge body of research in related fields that is generally applicable to drug R&D at NIBR. As NIBR develops a more complex IT infrastructure, the processes that accompany this infrastructure could greatly benefit from investment in additional operations focused research. Below are three specific in vivo research topics that could potentially be integrated into these automated tools if frameworks similar to the one discussed in this paper were to be developed:

Holding Room Assignments- Currently, room assignments are determined based on DA room allocation and historical room assignments. Once a scientist begins to use a room, they typically do not move their animals or research to a different area. In many holding areas within the vivariums, animals from one research project will occupy a single rack or even an entire room. This is highly inefficient and many times racks will only be partially filled with cages. In the future, room assignments should be based on overall research needs, including capacity and logistical concerns. Developing an automated algorithm for determining these assignments based on past, current and future needs would provide greater rack utilization, significantly increasing floor space efficiency (i.e. more animals per square foot).

Study Support and Training Coordination- Coordination of study support has been a historical problem in LAS. Similar to the issue discussed with special instructions (Chapter 2), many study

support requests involve detailed information and special processes. In general, scientists would like to have LAS provide more study support, but they are not comfortable with the current communication and scheduling process. LAS would benefit from the development of a tool that collected study support requests during the initial work request submission process. Once in the system, this data could be used to allocate study support resources. One of the challenges involved with this process is understanding which associates are approved to perform various procedures. To address this concern, a direct link to a training database is required.

LAS Staff Scheduling- LAS managers would like to transition to a more complex staffing model where weekend and night shifts are staffed. Currently there is no process for determining how these staffing decisions would be made. During the literature review process, I found several journal articles that discussed nurse duty scheduling. These processes used preference information entered by the nurses to develop schedules taking into account many staffing constraints. If LAS transitions to a more complex staffing model, they would benefit from having a tool that aligns research needs with organizational skills in an unbiased manner.

5.3. Conclusion

The pharmaceutical industry is experiencing significant competitive pressures. Innovation productivity continues to decline, while the costs for drug R&D steadily rise. This project was undertaken with the hope that improving research operations will both lower drug R&D costs and improve the efficiency of the R&D process. At the heart of these problems are organizational norms, which have created significant operational inefficiencies. Many of the communication processes in place today use ad hoc methods for relaying critical information between research teams and the vivarium staff. Addressing these problems will require a fundamental shift in the behavior of the organizations. A greater level of trust must be established between the research community and LAS, in order to drive better collaboration and high efficiencies. This goal cannot be achieved with the processes that exist today, which must be improved to produce reliable and robust results. Integral to this improvement effort is the development of a fair and robust method for allocating in vivo resources to research projects using a centralized management system.

This thesis provides the framework for developing a key portion of this management system. The architecture of this tool requires a web-based interface in order to provide seamless access to the research community. Based on research workflows, the proposed tool coordinates input from scientists and uses this information to schedule the required resources. Designing a tool that empowers the research community and provides an efficient means to manage projects is integral to implementing a successful solution. The complex constraints found in a research animal facility dictate the need for a unique scheduling algorithm. This thesis has explored these constraints in-depth and used an understanding of these challenges to develop a mixed integer linear programming model. A multi-criteria objective function uses the researcher's preference to

optimize both room assignments and procedure start times. A heuristic based algorithm using a tabu search strategy has been developed to provide a viable means for generating a near-optimal solution. Integrating these algorithms into a scheduling and coordination tool will begin the transformation that is required to meet the challenging operational goals that have been established.

This page has been intentionally left blank

6. Bibliography

Astrofsky, K. (2006, September). Risk Assessment of NIBRI- LAS Vivarium to Infectious Pathogen Introduction.

Ding, H., Lim, A., Rodrigues, B., & Zhu, Y. (2004). New heuristics for over-constrained flight to gate assignments. *Journal of the Operational Research Society, v 55, n 7* , 760-768.

Dorndorf, U., Drexl, A., Nikulin, Y., & Pesch, E. (2007). Flight gate scheduling: State-of-the-art and recent developments. *Omega* , 326.

Drexl, A., & Nikulin, Y. (2008). Multicriteria airport gate assignment and Pareto simulated annealing. *IIE Transactions (Institute of Industrial Engineers), v 40, n 4* , p 385-397.

European Federation of Pharmaceutical Industries and Associations. (2003). *The Pharmaceutical Industry in Figures- 2003 Update.* Brussels.

Glover, F. (2007). Tabu search--Uncharted domains. *Annals of Operations Research* , Vol. 149, Iss. 1, P 89-98.

Guinet, A., & Chaabane, S. (2003). Operating theatre planning. *International Journal of Production Economics, v 85* , 69-81.

Jebali, A., Hadj Alouane, A. B., & Ladet, P. (2006). Operating rooms scheduling. *International Journal of Production Economics, v 99, n 1-2, January/February, 2006, Control and Management of Productive Systems* , 52-62.

Lim, A., Rodrigues, B., & Zhu, Y. (2005). Airport gate scheduling with time windows. *Artificial Intelligence Review, v 24* , 5-31.

National Research Council. (2002). *Guide for the Care and Use of Laboratory Animals.* Washington D.C.: National Academy Press.

Novartis AG. (2008). *2007 Annual Report.* Novartis.

Ruys, T. (1991). *Handbook of Facilities Planning, Volume 2: Laboratory Animal Facilities.* New York: Van Nostrand Reinhold.

Standard & Poor's. (2008). *Industry Surveys- Healthcare: Pharmaceuticals.*

The Congress of the United States- Congressional Budget Office. (October 2006). *Research and Development in the Pharmaceutical Industry.*

US Department of Agriculture. (2005, November). Animal Welfare Act and Animal Welfare Regulations.

Xu, J., & Bailey, G. (2001). The Airport Gate Assignment Problem: Mathematical model and a tabu search algorithm. *Proceedings of the Hawaii International Conference on System Sciences* , 77.

7. Appendix A- Subroutine Algorithm Detail

7.1. Compatibility Check Subroutine

CompatCheck(j, l) is short for *Compatibility Check*. This algorithm evaluates the compatibility of job j, currently assigned to procedural space i, with a potential move to procedural space l, where $i \neq l$. Specifically, this algorithm determines if the species for job j is compatible with the species restrictions for procedural space l. In addition, this algorithm determines if procedural space l contains the equipment necessary to perform job j. These comparisons are similar to the constraints discussed in Chapter 3 and detailed in Equations (11), (12) and (14). If these conditions are successfully met, *CompatCheck(j, l)* returns as value of "true". Otherwise, the algorithm returns a value of "false", which indicates the job is not compatible with procedural space l.

7.2. Shift Subroutines

ShiftRight(j, i, τ) shifts the start time of job j assigned to procedural space i. The magnitude of this rightward shift shall be τ time units. When using this subroutine, it is critical that the value specified for τ will not cause overlap with adjacent jobs and will not violate the upper scheduling bound restriction (i.e. standard operating hours).

AttemptShiftRight(j, i) returns the latest time that job j can begin in procedural space i. This algorithm only evaluates moving job j and not additional jobs that are downstream from job j. The value returned will either be constrained by the assigned start time of a later job or the upper scheduling bound. Note that this subroutine does not actually modify the schedule, but only returns a value related to the possible move. Once a potential time shift value is identified (τ), this algorithm executes the *RoomCheck(j, l, τ)* subroutine to ensure species conflicts do not occur for spaces that share a common room. The τ parameter for this function will equal the current assigned start time for job j plus the time shift value. If *RoomCheck(j, l, τ)* returns a value of "false", then the *AttemptShiftRight(j, i)* returns a value of zero. Otherwise, the *AttemptShiftRight(j, i)* function returns the shift value.

ShiftLeft(j, i, τ) shifts the start time of job j assigned to procedural space i. The magnitude of this leftward shift shall be τ time units. When using this subroutine, it is critical that the value specified for τ will not cause overlap with adjacent jobs and will not violate the lower scheduling bound restriction (i.e. standard operating hours).

AttemptShiftLeft(j, i) returns the earliest time that job j can begin in procedural space i. This algorithm only evaluates moving job j and not additional jobs that are upstream from job j. The value returned will either be constrained by the assigned start time of an earlier job or the lower scheduling bound. Note that this subroutine does not actually modify the schedule, but only

returns a value related to the possible move. Once a potential time shift value is identified (τ), this algorithm executes the *RoomCheck(j, l, τ)* subroutine to ensure species conflicts do not occur for spaces that share a common room. The τ parameter for this function will equal the current assigned start time for job *j* minus the time shift value. If *RoomCheck(j, l, τ)* returns a value of "false", then the *AttemptShiftLeft(j, i)* returns a value of zero. Otherwise, the *AttemptShiftLeft(j, i)* function returns the shift value.

7.3. Optimization Subroutines

GapMeasure(j, i) returns the difference between the assigned start time and requested start time for job *j* in procedural space *i*. This comparison is written as $t_j - \omega_j$. A positive value indicates the job is assigned a starting time that is later than the requested start time. Conversely, a negative value indicates that the job is assigned a start time that is earlier than the requested start time.

OptimizeRight(j, i) attempts to shift the assigned starting times beginning at job *j* in procedural space *i* to the right (later), with a goal of reducing the difference between the assigned and requested start times. The algorithm uses the *GapMeasure(j, i)* subroutine to determine the magnitude of difference between the start time values. Job *j* is not moved if *GapMeasure(j, i)* returns a positive value. However, the *AttemptShiftRight(j, i)* subroutine will be invoked if a negative value is returned. Notice that the *RoomCheck(j, l, τ)* subroutine verifies procedural space compatibility when the *AttemptShiftRight(j, i)* subroutine is executed. If *AttemptShiftRight(j, i)* returns a value greater than zero, the *ShiftRight(j, i, τ)* subroutine is used to move the assigned start time for job *j* closer to the requested start time. The time shift value *(τ)* will equal either the absolute value of *GapMeasure(j, i)* or *AttemptShiftRight(j, i)*, whichever value is smaller. In the event that the *GapMeasure(j, i)* value is chosen, the *AttemptShiftRight(j, i)* algorithm is ran again to ensure that no species conflicts exist. The output of this function will then be used to adjust the assigned start time. This algorithm performs the same evaluation and move process for all jobs that are scheduled to the left (earlier) of job *j*, moving from the latest scheduled job to the earliest scheduled job.

OptimizeLeft(j, i) attempts to shift the assigned starting times beginning at job *j* in procedural space *i* to the left (earlier), with a goal of reducing the difference between the assigned and requested start times. The algorithm uses the *GapMeasure(j, i)* subroutine to determine the magnitude of difference between the start time values. Job *j* is not moved if *GapMeasure(j, i)* returns a negative value. However, the *AttemptShiftLeft(j, i)* subroutine will be invoked if a positive value is returned. Notice that the *RoomCheck(j, l, τ)* subroutine verifies procedural space compatibility when the *AttemptShiftLeft(j, i)* subroutine is executed. If *AttemptShiftLeft(j, i)* returns a value greater than zero, the *ShiftLeft(j, i, τ)* subroutine is used to move the assigned start time for job *j* closer to the requested start time. The time shift value *(τ)* will equal either the absolute value of *GapMeasure(j, i)* or *AttemptShiftLeft(j, i)*, whichever value is smaller. In the

event that the *GapMeasure(j, i)* value is chosen, the *AttemptShiftLeft(j, i)* algorithm is ran again to ensure that no species conflicts exist. The output of this function will then be used to adjust the assigned start time. This algorithm performs the same evaluation and move process for all jobs that are scheduled to the right (later) of job *j*, moving from the earliest scheduled job to the latest scheduled job.

7.4. Time Point Subroutines:

PrevEnd(j) returns the ending time (i.e. start time plus the job duration) for the job that is scheduled just prior to (earlier) job *j*.

NextStart(j) returns the starting time for the job that is scheduled just after (later) job *j*.

7.5. Interval Subroutines

An interval is a contiguous block of time that is bounded by two distinct time points.

PrevJob(j) returns the job that precedes job *j*.

NextJob(j) returns the job that follows job *j*.

InitializeInterval(X, j) creates an interval that is defined by the start and end times associated with job *j*. The variable *X* represents the name of this interval. The interval created using the *InitializeInterval(X, j)* function will be identical to the job used to define the interval.

ExtendRight(X) increases the size of interval *X* by changing the end point of the interval to a later time. This new time point is the ending time of the following job.

AttemptExtendRight(X) returns a Boolean value that indicates the ability to perform an *ExtendRight(X)* operation for interval *X*. A value of "true" indicates that the operation can be accommodated (i.e. another job exists after job *j*) and a value of "false" indicates that the operation will not be successful.

ExtendLeft(X) increases the size of interval *X* by changing the end point of the interval to an earlier time. This new time point is the starting time of the preceding job.

AttemptExtendLeft(X) returns a Boolean value that indicates the ability to perform an *ExtendLeft(X)* operation for interval *X*. A value of "true" indicates that the operation can be accommodated (i.e. another job exists prior to job *j*) and a value of "false" indicates that the operation will not be successful.

This page has been intentionally left blank

8. Appendix B- Insert Move Algorithm

Algorithm 1- The Insert Move Algorithm

success ← *false*
while not success **do**
randomly select job *j* and targeted procedural space *l*
determine value of originating procedural space *i*
CompatCheck(j, l)
if *CompatCheck(j, l)* **then**
offset ← 24 hours (i.e. a large number)
jobtarget ← empty
for each job *v* in procedural room *l* **do**
$u \leftarrow$ *PrevJob(v)*
$t_m \leftarrow \frac{(t_u+\Delta_u+t_v)}{2}$ {calculates the median time between jobs *u* and *v*}
if *offset* > $|t_m - \omega_j|$ **then**
offset ← $|t_m - \omega_j|$
jobtarget ← job *v*
end if
end for
$u \leftarrow$ *PrevJob(v)*
$t_2 \leftarrow$ the assigned start time of *jobtarget*
$t_1 \leftarrow$ the end time of the job that precedes *jobtarget*
if $t_2 - t_1 \geq \Delta_j$ **then**
move ← *true*
for all shared procedural spaces ($X_{il} = 1$) **do**
if not *RoomCheck(j, l, t_1)* **then**
move ← *false*
end if
end for
if *move* **then**
success ← *true*
remove job *j* from procedural space *i*
insert job *j* into procedural space *l*, with $t_j = t_1$
OptimizeRight(j, l)
end if
end if
$t_3 \leftarrow$ *AttemptShiftLeft(u, l)*
$t_4 \leftarrow$ *AttemptShiftRight(v, l)*
if $t_2 - t_1 < \Delta_j$ **and if** $\Delta_j < (t_2 - t_1) + t_3 + t_4$ **then**
ShiftLeft(u, l, t_3)
ShiftRight (v, l, t_4)
$t_5 \leftarrow t_3 + \Delta_u$
move ← *true*
for all shared procedural spaces ($X_{il} = 1$) **do**
if not *RoomCheck(j, l, t_5)* **then**
move ← *false*
end if
end for
if *move* **then**
success ← *true*
remove job *j* from procedural space *i*
insert job *j* into procedural space *l*, with $t_j = t_5$
OptimizeLeft(j, l)

```
                    OptimizeRight(j, l)
                end if
            end if
        end if
    end while
    if move then
    OptimizeLeft(v, i) {where v is the job that originally followed job j}
    OptimizeRight(u, i) {where u is the job that originally proceeded job j}
    else output "Insert Move Failed";
```

9. Appendix C- Interval Exchange Move Algorithm

Algorithm 2- The Interval Exchange Move Algorithm

success ← *false*
while not success **do**
randomly select job *j* and targeted procedural space *l*
determine value of originating procedural space *i*
CompatCheck(j, l)
if *CompatCheck(j, l)* **then**
exchange ← *false*
InitializeInterval(A, j)
k← earliest job in procedural space *l*, 0 if no jobs scheduled for procedural space *l*
if $k \neq 0$ **then**
$t_{11} \leftarrow PrevEnd(j)$
$t_{12} \leftarrow t_j$
$t_{13} \leftarrow t_j + \Delta_j$
$t_{14} \leftarrow NextStart(j)$
while not exchange **do**
$t_{21} \leftarrow PrevEnd(k)$
$t_{22} \leftarrow t_k$
$t_{23} \leftarrow t_k + \Delta_k$
$t_{24} \leftarrow NextStart(k)$
if $t_{12} \leq t_{23} \leq t_{13}$ **then**
exchange ← *true*
else if $t_{12} \leq t_{22} \leq t_{13}$ **then**
exchange ← *true*
else
$k \leftarrow NextJob(k)$
end while
InitializeInterval(B, k)
if *RoomCheck(j, l,* t_{12}*)* **and if** *RoomCheck(k, i,* t_{22}*)* **then**
align ← *false*
execute ← *true*
a ← *job j*
b ← *job j*
u← *job k*
v ← *job k*
while not *aligned* **and while** *execute* **do**
align ← *true*
if $t_{21} > t_{12}$ **then**
if *AttemptExtendLeft(B)* **then**
$u \leftarrow PrevJob(u)$
$t_{21} \leftarrow PrevEnd(u)$
$t_{22} \leftarrow t_u$
if *RoomCheck(u, i,* t_{22}*)* **then**
ExtendLeft(B)
aligned ← *false*
else
execute ← *false*
else
execute ← *false*
end if
if $t_{11} > t_{22}$ **then**
if *AttemptExtendLeft(A)* **then**
$a \leftarrow PrevJob(a)$

```
                        t11 ← PrevEnd(a)
                        t12← ta
                        if RoomCheck(a, l, t12) then
                            ExtendLeft(A)
                            aligned ← false
                        else
                            execute ← false
                    else
                        execute ← false
                end if
                if t23 > t14 then
                    if AttemptExtendRight(A) then
                        b ←NextJob(b)
                        t13 ← tb+Δb
                        t14 ← NextStart(b)
                        if RoomCheck(b, l, tb) then
                            ExtendRight(A)
                            aligned ← false
                        else
                            execute ← false
                    else
                        execute ← false
                end if
                if t13 > t24 then
                    if AttemptExtendRight(B) then
                        v←NextJob(v)
                        t23 ← tv+Δv
                        t24 ← NextStart(v)
                        if RoomCheck(v, i, tv) then
                            ExtendRight(B)
                            aligned ← false
                        else
                            execute ← false
                    else
                        execute ← false
                end if
            end while
        end if
    else
        if RoomCheck(j, l, tj) then
            execute ← true
        end if
        if execute then
            success ← true
            exchange intervals A and B
            OptimizeLeft(a, l)
            OptimizeRight(b, l)
            OptimizeLeft(u, i)
            OptimizeRight(v, i)
        end if
    end if
end while
```