Distributed Operating Room Scheduling using Lazy Constraints and Networks

Ethan Merrick[1], Benjamin Solomon[1] and Mitchell Clark[1]

[1]The University of Queensland

Abstract

We re-implemented existing models[1] for solving the distributed operating room scheduling (DORS) problem. We extend the existing formulation with a new LBBD cut, we extend the implementation by employing lazy constraints in Gurobi and we try a network model formulation. We were not able to replicate results in line with the original paper. It was found that... <TO BE CONTINUED>.

1 Introduction

Our report is based upon and summarises Roshanaei, Luong, Aleman and Urbach [1] which we will also refer to as *the original paper* throughout. The paper solves the deterministic distributed operating room scheduling (DORS) problem using Logic-Based Benders' Decomposition (LBBD) and a cut propagation method. This formulation has been extended upon in a stochastic setting[2]. We re-implemented the paper and extended the formulation with a new LBBD cut and a network model. We extended implementation by using lazy constraints in Gurobi. Our main goals were to replicate the results found in the original paper and investigate if there was potential improvement to be had by using lazy constraints instead of iteration.

The LBBD framework outlined in [1] is structured as a location-allocation master problem with a bin-packing sub problem simplified by symmetric operating rooms. The results in the original paper show that the proposed LBBD models scale well for up to 160 patients.

The original paper contains many variants of models, they provide a pure MIP formulation, and an LBBD formulation with three different types of cuts. For each of these cuts they experiment with a version with and without propagation. For each of these they implement different cut generation schemes which determine how many sub problems to iterate over before re-solving the master problem. We choose some of their models to best achieve our goals. We choose to use the maximal cut generation scheme in all of our LBBD models, iterating over all sub-problems before re-solving the master problem.

In the DORS problem, we seek to decide which hospital operating suites and their respective operating rooms to open. We also decide which patients to allocate to each hospital day. We make these decisions in such a way as to minimize the the total cost of opening facilities while also trying to maximize the reward for assigning critical patients[1].

2 Model Formulation

2.1 Master Problem

The following is the LBBD formulation of the problem. For a pure IP formulation refer to [1]. We will first outline the formulation of the master problem. The master problem handles assignment of patients to hospital-days.

SETS

$\mathcal{P}$	Set of patients $p \in \mathcal{P}$
$\mathcal{P}'$	Set of mandatory patients, $\mathcal{P}' = \{p \mid \rho_p(\lvert\mathcal{D}\rvert - \alpha_p) \leq -\Gamma\}$
$\mathcal{H}$	Set of hospitals, $h \in \mathcal{H}$
$\mathcal{D}$	Set of days in the planning horizon, $d \in \mathcal{D}$
$\mathcal{R}_h$	Set of ORs in each hospital's surgical suite, $r \in \mathcal{R}_h$

DATA

G_{hd}	Cost of opening the surgical suite in hosptial h on day d.
F_{hd}	Cost of opening and OR in hospital h on day d.
B_{hd}	Regular operating hours of each OR on day d. in hospital h.
T_{hp}	Total booked time (preparation time + surgery time + cleaning time) of patient p.
ρ_p	Health status score assigned to patient p.
α_p	Number of days elapsed from the referal date of patient p.
κ_1	Waiting cost for scheduled patients.
κ_2	Waiting cost for unscheduled patients.
Γ	Health status threshold above which patients have to be operated.

VARIABLES

x_{hdp}	1 if patient p is assigned to hospital h on day d , 0 otherwise
u_{hd}	1 if the surgical suite in hospital h is opened on day d, 0 otherwise
y_{hd}	$\in \mathbb{Z}^+$, lower bound on number of operating rooms open in hospital h on day d
w_p	1 if patient p is not scheduled this horizon, 0 otherwise

Objective The objective function balances the minimisation of costs associated with opening hospitals and ORs and maximising the reward of assigning patients to surgeries.

$$\begin{aligned}\text{minimize}\Bigg(&\sum_{h\in\mathcal{H}}\sum_{d\in\mathcal{D}} G_{hd}U_{hd} + \sum_{h\in\mathcal{H}}\sum_{d\in\mathcal{D}} F_{hd}y_{hd} \\ &+ \sum_{h\in\mathcal{H}}\sum_{d\in\mathcal{D}}\sum_{p\in\mathcal{P}} \kappa_1[\rho_p(d-\alpha_p)x_{hdp}] \\ &+ \sum_{p\in\mathcal{P}\backslash\{\mathcal{P}'\}} \kappa_2[\rho_p(\mathcal{D}+1-\alpha_p)w_p]\Bigg)\end{aligned} \tag{1}$$

Constraints The constraints for the MP are formulated as follows.

$$\sum_{h\in\mathcal{H}}\sum_{d\in\mathcal{D}} x_{hdp} = 1 \qquad \forall p\in\mathcal{P}' \tag{2}$$

$$\sum_{h\in athcalH}\sum_{d\in\mathcal{D}} x_{hdp} + w_p = 1 \qquad \forall p\in\mathcal{P}\backslash\{\mathcal{P}'\} \tag{3}$$

$$x_{hdp} \le u_{hd} \quad \forall h\in\mathcal{H}, d\in\mathcal{D}, p\in\mathcal{P} \tag{4}$$

$$\sum_{p\in\mathcal{P}} T_p x_{hdp} \le |\mathcal{R}_h| B_{hd} u_{hd} \qquad \forall h\in\mathcal{H}, d\in\mathcal{D} \tag{5}$$

$$T_p\, x_{hdp} \le B_{hd} \quad \forall h\in\mathcal{H}, d\in\mathcal{D}, p\in\mathcal{P} \tag{6}$$

$$\frac{\sum_{p\in\mathcal{P}} T_p x_{hdp}}{B_{hd}} \le y_{hd} \qquad \forall h\in\mathcal{H}, d\in\mathcal{D} \tag{7}$$

$$y_{hd} \le |\mathcal{R}_h| \qquad \forall h\in\mathcal{H}, d\in\mathcal{D} \tag{8}$$

$$u_{hd}, x_{hdp} \in \{0,1\} \quad \forall h\in\mathcal{H}, d\in\mathcal{D}, p\in\mathcal{P} \tag{9}$$

$$w_p \in \{0,1\} \qquad \forall p\in\mathcal{P}\backslash\{\mathcal{P}'\} \tag{10}$$

Constraint (2) ensures all mandatory patients are assigned in the planning horizon. Constraint (3) ensures that variables x_{hdp} and u_{hd} are not turned on simultaneously. Constraint (4) ensures that if a patient is assigned a hospital-day then that hospital-day is open. (5) ensures that surgery time of patients assigned to a hospital-day does not exceed the available surgery time in that hospital day. (6) ensures an individuals surgery time does not exceed the hospital-day's available time. (7) ensures y_{hd} gives a lower bound on the number of operating rooms. (8) ensures y_{hd} does not exceed the number of operating rooms available on a hospital-day. Constraints (9) – (10) simply restrict variables to binary.

2.2 Subproblems

Given a solution to the master problem $(\widehat{Y}^{(i)}_{hd}, \widehat{\mathcal{P}}^{(i)}_{hd})$ the subproblem minimises the number of ORs to open for a given hospital day. Each subproblem is formulated as follows.

ADDITIONAL VARIABLES

y_r $\in \mathbb{Z}^+$, number of open operating rooms.
x_{pr} 1 if patient p is assigned to operating room r, 0 otherwise.

$$\text{minimise} \quad \overline{Y}_{hd} = \sum_{r\in\mathcal{R}_h} y_r \tag{11}$$

With constraints given as follows:

$$\sum_{r\in\mathcal{R}_h} x_{pr} = 1 \qquad \forall p\in\widehat{\mathcal{P}}^{(i)}_{hd} \tag{12}$$

$$\sum_{p\in\widehat{\mathcal{P}}^{(i)}_{hd}} T_p x_{pr} \le B_{hd} y_r \qquad \forall r\in\mathcal{R}_h \tag{13}$$

$$x_{pr} \le y_r \qquad \forall p\in\widehat{\mathcal{P}}^{(i)}_{hd}, r\in\mathcal{R}_h \tag{14}$$

$$y_r \le y_{r-1} \qquad \forall r\in\mathcal{R}_h\backslash\{1\} \tag{15}$$

$$x_{pr},\, y_r \in \{0,1\} \qquad \forall p\in\widehat{\mathcal{P}}^{(i)}_{hd}, r\in\mathcal{R}_h \tag{16}$$

Constraint (12) ensures that each patient is assigned to only one operating room. Constraint (13) ensures that no OR is overcapacitated. Constraint (14) ensures that patients are assigned to open ORs. Constraint (15) breaks symmtry among ORs.

2.3 Benders Cuts

There are multiple forms of benders cuts outlined in the original paper. We will discuss pertinent forms based on their performance as discussed in the original paper. These are the LBBD1 and LBBD2 benders cuts. In order to describe these cut types, we will first discuss the first-fit decreasing heuristic algorithm (FFD) as this is used to determine optimality of SPs.

First-fit decreasing heuristic algorithm Since the SP packing problem can be difficult to solve, we can first find a feasible solution $(\overline{F}^{(i)}_{hd})$ using a FFD heuristic. This process is faster than other techniques such as integer and constraint programming. Moreover, we have the following relationship between the FFD, MP and SP solutions;

$$\tilde{Y}^{(i)}_{hd} \le \overline{Y}^{(i)}_{hd} \le \overline{F}^{(i)}_{hd} \tag{17}$$

We can use the FFD solution $\left(\overline{F}^{(i)}_{hd}\right)$ to find an optimal SP solution $\left(\overline{Y}^{(i)_{hd}}\right)$ without explicitly solving the SP. Moreover, if $\overline{F}^{(i)}_{hd} \neq \tilde{Y}^{(i)}_{hd}$ then when solving the SP we can use $\min\{\overline{F}^{(i)}_{hd}, |\mathcal{R}_h|\}$ as an upper bound.

LBBD1 LBBD1 [1] utilises both feasibility and optimality cuts to correct the MP to find a solution. If the SP is infeasible the following "no good" cut is added to the MP

which requires at least one patient be removed from $\tilde{\mathcal{P}}_{hd}^{(i)}$.

$$\sum_{p \in \tilde{\mathcal{P}}_{hd}^{(i)}} (1 - x_{hdp}) \geq 1 \qquad \forall (h,d) \in \mathcal{U}_{hd}^{(i)} \qquad (18)$$

Where $\mathcal{U}_{hd}^{(i)}$ is the set of infeasible SPs at this stage of solving the MP. If the SP is optimal, that is $\tilde{Y}_{hd}^{(i)} = \overline{Y}_{hd}^{(i)}$, no cuts are required. However, if the this is not the case, the following optimality cut is added:

$$y_{hd} \geq \overline{Y}_{hd}^{(i)} - \sum_{p \in \hat{\mathcal{P}}} (1 - x_{hdp}) \qquad \forall (h,d) \in \overline{\mathcal{J}}^{(i)}.$$

Where $\overline{\mathcal{J}}^{(i)}$ is the set of SPs that are not optimal at a particular stage of solving the MP. This cut effectively has two results. It either forces at least one more OR open, or removes one patient from $\tilde{\mathcal{P}}_{hd}^{(i)}$.

LBBD2 LBBD2 differs from LBBD1 in its feasibility cut which is given as follows:

$$y_{hd} \geq (|\mathcal{R}_h| + 1) - \sum_{p \in \hat{\mathcal{P}}_{hd}^{(i)}} (1 - x_{hdp}) \quad (h,d) \in \overline{\mathcal{U}}^{(i)}.$$

If $\overline{Y}_{hd}^{(i)} = |\mathcal{R}_h|$ and the SP is infeasible then this cut simply removes one patient from $\overline{\mathcal{P}}_{hd}^{(i)}$. However, if $\hat{Y}_{hd}^{(i)} \leq |\mathcal{R}_h$ and the SP is infeasible then the cut either removes two patients from $\overline{\mathcal{P}}_{hd}^{(i)}$, or removes one patient and/or opens at least one more OR.

2.4 Cut Propagation

The original paper utilises cut propagation for LBBDs in order to generate multiple cuts for each infeasible SP. This is done by recognising that an infeasible set of patients for a particular hospital-day cannot be packed into a hospital-day with less or equal OR time.

2.5 Network Problem

We also give the formulation of the problem as a network. The problem contains the sets and data of the master problem previously outlined with the addition of the following sets and data.

SETS

$\mathcal{N}$	Set of nodes $n \in \mathcal{N}$
$\mathcal{N}'$	Set of nodes $n \in \mathcal{N}$ s. t. minDur $\leq n[time] \leq B_{n[time],n[day]} -$ minDur
$\mathcal{A}$	Set of arcs $a \in \mathcal{A}$

DATA

t_n	Arcs that enter node $n \in \mathcal{N}$
f_n	Arcs that leave node $n \in \mathcal{N}$
minDur	minimum surgery duration

The variables are also shared with the master problem with the addition of the following variable.

VARIABLES

z_a	1 if arc a is turned on, 0 otherwise.

Objective The objective function is identical to master problem objective given in 1.

$$\text{minimize} \Bigg(\sum_{h \in \mathcal{H}} \sum_{d \in \mathcal{D}} G_{hd} U_{hd} + \sum_{h \in \mathcal{H}} \sum_{d \in \mathcal{D}} F_{hd} y_{hd} \qquad (19)$$

$$+ \sum_{h \in \mathcal{H}} \sum_{d \in \mathcal{D}} \sum_{p \in \mathcal{P}} \kappa_1 [\rho_p (d - \alpha_p) x_{hdp}] \qquad (20)$$

$$+ \sum_{p \in \mathcal{P} \setminus \{\mathcal{P}'\}} \kappa_2 [\rho_p (\mathcal{D} + 1 - \alpha_p) w_p] \Bigg)$$

Constraints NOT FINISHED The constraints for the network formulation are given as follows,

$$\sum_{a \in f_n} z_a = \sum_{a \in t_n} z_a \qquad \forall n \in \mathcal{N} \qquad (21)$$

$$\sum_{a \in \mathcal{A}} \qquad (22)$$

$$\sum_{h \in \mathcal{H}} \sum_{d \in \mathcal{D}} x_{hdp} = 1 \qquad \forall p \in \mathcal{P}' \qquad (23)$$

$$\sum_{h \in athcal H} \sum_{d \in \mathcal{D}} x_{hdp} + w_p = 1 \qquad \forall p \in \mathcal{P} \setminus \{\mathcal{P}'\} \qquad (24)$$

$$x_{hdp} \leq u_{hd} \quad \forall h \in \mathcal{H}, d \in \mathcal{D}, p \in \mathcal{P} \qquad (25)$$

$$\sum_{p \in \mathcal{P}} T_p x_{hdp} \leq |\mathcal{R}_h| B_{hd} u_{hd} \qquad \forall h \in \mathcal{H}, d \in \mathcal{D} \qquad (26)$$

$$T_p \, x_{hdp} \leq B_{hd} \quad \forall h \in \mathcal{H}, d \in \mathcal{D}, p \in \mathcal{P} \qquad (27)$$

$$\frac{\sum_{p \in \mathcal{P}} T_p x_{hdp}}{B_{hd}} \leq y_{hd} \qquad \forall h \in \mathcal{H}, d \in \mathcal{D} \qquad (28)$$

$$y_{hd} \leq |\mathcal{R}_h| \qquad \forall h \in \mathcal{H}, d \in \mathcal{D} \qquad (29)$$

$$u_{hd}, x_{hdp} \in \{0,1\} \quad \forall h \in \mathcal{H}, d \in \mathcal{D}, p \in \mathcal{P} \qquad (30)$$

$$w_p \in \{0,1\} \qquad \forall p \in \mathcal{P} \setminus \{\mathcal{P}'\} \qquad (31)$$

Constraint (23) ensures all mandatory patients are assigned in the planning horizon. Constraint (24) ensures that variables x_{hdp} and u_{hd} are not turned on simultaneously. Constraint (25) ensures that if a patient is assigned a hospital-day then that hospital-day is open. (26) ensures

that surgery time of patients assigned to a hospital-day does not exceed the available surgery time in that hospital day. (27) ensures an individuals surgery time does not exceed the hospital-day's available time. (28) ensures y_{hd} gives a lower bound on the number of operating rooms. (29) ensures y_{hd} does not exceed the number of operating rooms available on a hospital-day. Constraints (30) – (31) simply restrict variables to binary.

3 Data

We generate our own data based on the parameters provided in the original paper. This data is derived from 7500 elective and emergency patients from 2011 to 2013 in the General Surgery Departments of the University Health Network (UHN) hospitals (Toronto, Ontario, Canada) by considering the average number of surgical cases in a week[1]. The original instances used were not available. Using examples given by Guo[2] we round surgery times to integer values. We generated data according to the distributions in Table 1.

Table 1: *Distributions used for data generation.*

Data	Distribution
κ_1	50 dollars
κ_2	5 dollars
Γ	500
ρ_p	Discrete uniform distribution [1, 5]
B_{hd}	Discrete uniform distribution [420, 480] minutes in 15-minute intervals
α_p	Discrete uniform distribution [60, 120] days.
F_{hd}	Discrete uniform distribution [4000, 6000] dollars
G_{hd}	Discrete uniform distribution [1500, 2500] dollars
T_p	Truncated normal distribution [45, 480] minutes, $\mu = 160$, $\sigma = 40$

4 Implementation

We implemented two different practical models for LBBD, one following [1] which solves the master problem iteratively in a loop, rebuilding the master problem branch-and-bound tree on every iteration. The other model leveraged lazy constraints in Gurobi to perform a branch and check routine[3], building the branch-and-bound tree once while applying Benders' cuts as lazy constraints at each new incumbent solution. Both implementatoins of LBBD first solved each sub problem using the first-fit decreasing (FFD) heuristic[4] and used its results as a warm-start to a MIP model as outlined in [1]. The pure MIP and network models were solved using standard Gurobi routines.

Model formulations were simple enough to interpret but practical implementation was difficult for a few reasons. Firstly, the original paper did not supply any code. This made it hard to find implementation details that may have drastically improved results, for example some important caching protocol may have been used that was not mentioned in the paper. Further, the paper supplied distribution parameters that were useful in generating data, however it did not supply the data instances that were used to generate their results. This made it impossible to verify if the differences seen in our results were due to an implementation error, random chance or something else. It also left data inadequately described. For example the surgery times were said to be from a normal distribution, this might imply that they should be real-valued. However, upon inspection of another paper's data which worked on a similar problem[2], it was noticed that surgery times were rounded to some integer value. Implementing this with our data generation scheme immediately improved the performance of our models. We reached out to the author of the original paper and we were told that they were no longer in possession of either the code or data that was used.

The paper did not supply example objective values found for their instances which made it hard to determine feasibility of our solutions. The original paper also did not specify if a certain MIP gap parameter was set, it mentions "Gurobi optimal solutions" but by default Gurobi treats a MIP gap of 1% as optimal, [2] states that they solve their models to "optimality with a gap of 1%" when they seek to make comparisons with the original paper, meaning that the exact gap used in the original paper was left unclear to us. Ultimately the main verification method of our results was by the consistency of outputted optimal objective values across all models implemented. That is, the pure MIP, Network and Benders' models were implemented in sufficiently different ways but all provided the same output giving us some level of confidence in our implementation's optimality. Feasibility was verified by manual inspection of patient allocations for instances with small number of patients. We had no way of verifying if the speed of our models would be the same as achieved by the models used in the original paper.

5 Results

Models were run on a Windows 11 computer with a AMD Ryzen 5 5625U 2.3GHz processor with 16 GB of RAM. Gurobi version 10.0.1 was used with Python version 3.9.16.

We aimed to compare 7 models: The pure MIP, the network model, an iterative LBBD1 (iLBBD1), LBBD1 using lazy constraints (cLBBD1), iterative LBBD2 with propagation (iLBBD2p), LBBD2 with lazy constraints and propagation (cLBBD2p) and LBBD4 with lazy constraints and propagation (cLBBD4). The pure MIP was used as a baseline. iLBBD1 and cLBBD1 were chosen to be used as baseline LBBD models. LBBD2 with propagation was chosen as it was one of the best performers

in the original paper, we intended to use it as the main comparison to the network model and our new cut. We chose to collect results for LBBD4 only using propagation and lazy constraints as we believed this would give us the best results while adhering to the project time constraints.

We generate 5 seeded instances of data similarly to [1]. The data is generated for 3 hospitals and 5 operating rooms over a 5 day planning horizon for varying number of patients. Unlike [1] who generate a set of data for 3 and 5 operating rooms while only varying the surgery times in each instance, we generate each instance with entirely different data. This was believed to give results more indicative of a wider variation in problem parameters. Also unlike in the original paper, where models were ran with a time limit of 7200 seconds, we set our time limit to 900 seconds due to project time constraints. At a time limit of 7200 seconds the worst case time to run all models over all 5 instances was 70 hours for only one patient size. For this same reason we only ran models with 5 available operating rooms, instead of testing 5 and 3. We chose to use 5 operating rooms, this should have led to harder sub problems and relatively easier master problems compared to using 3 operating rooms[1].

There was uncertainty as to whether the models were solved to true optimality or to a relative MIP gap of 1% as was done by [2], so we report results for both scenarios. We follow [1] by defining the best performing model to be the most robust, that is, the one able to solve the model to optimality within the given time constraints while breaking ties by considering the fastest models averaged over solved instances.

The 7 models were solved to optimality on each of the 5 instances, the average time to solve can be seen in Table 2 and the average gaps can be seen in Table 3.

We can see from Table 2 that all models struggled solving to optimality for number of patients greater than only 20. The pure MIP is the most robust across all patient sizes while the Network model is the least. We can see that for sizes where LBBD models do solve an instance, they do so on average faster than the pure MIP. For example, for a patient size of 20 cLBBD4p performed the best, being solved in the least time on average, with cLBBD2p being close second. For a patient size of 60, most LBBD models solved faster than the pure MIP. Comparing between LBBD models, all using lazy constraints outperform those using iteration in terms of time to solve for every number of patients. The results shown in Table 2 suggest that none of the 7 models scale well for solving problems with sets of patients as large or larger than 60 when solving to optimality. No models were able to solve instances with 80 patients to optimality.

Table 3 shows that although not many instances solved to optimality, by the end of the prescribed time limit, the optimality gaps were very small. All optimality gaps had reached at least 1% or lower. In fact, increasing the number of patients did not appear to have a large negative effect on the final optimality gaps, giving merit to the scalability of all models in term of achieving a small gap in under 15 minutes for increasing number of patients. Interestingly, nearly all gaps for number of patients greater than 20 were of the same magnitude, it seems there is a is some threshold that the model can reach very quickly but closing the last slice of gap becomes very difficult. These results gave more motivation to investigate solving to a 1% gap.

The 7 models were solved to a 1% gap on each of the 5 instances, the average time to solve can be seen in Table 4 and the average gaps can be seen in Table 4.

Table 2: *Average time (seconds) until solved to optimality over 5 instances. The number of instances not solved to optimality are superscripted. Non-solved instances are not included in average. Asterisks indicate that no instances solved in time.*

$\|\mathcal{P}\|$	Pure MIP	Network	iLBBD1	cLBBD1	iLBBD2p	cLBBD2p	cLBBD4p
20	16.06	$****^{(5)}$	1.509	0.8829	1.431	0.8800	0.7890
40	$179.3^{(2)}$	$268.5^{(3)}$	$4.429^{(4)}$	$1.911^{(4)}$	$4.503^{(4)}$	$1.911^{(4)}$	$1.959^{(4)}$
60	$30.80^{(4)}$	$****^{(5)}$	$24.46^{(4)}$	$10.73^{(4)}$	$34.61^{(4)}$	$21.46^{(4)}$	$25.80^{(4)}$
80	$****^{(5)}$	$****^{(5)}$	$****^{(5)}$	$****^{(5)}$	$****^{(5)}$	$****^{(5)}$	$****^{(5)}$

Table 3: *Average gap (%) over 5 instances after trying to solve to optimality. MIPGap is reported for pure MIP, Network and callback implementations of LBBD. Gap between master problem lowerbound and best sub problem upperbound is report for iterative implementations of LBBD.*

$\|\mathcal{P}\|$	Pure MIP	Network	iLBBD1	cLBBD1	iLBBD2p	cLBBD2p	cLBBD4p
20	0.000	0.4950	0.000	0.000	0.000	0.000	0.000
40	0.04220	0.3382	0.6252	0.3532	0.09500	0.4765	0.4948
60	0.1827	0.4131	0.5514	0.2540	0.1700	0.3513	0.3273
80	0.2630	0.2048	0.9390	0.6000	0.2100	0.4718	0.5218

Table 4: *Average time (seconds) until solved to optimality with a 1% gap over 5 instances. The gap used to terminate optimisation was the MIPGap for all models except for iterative LBBDs which were terminated by gap between master problem and sub problem. The number of instances not solved to optimality are superscripted. Non-solved instances are not included in average. Asterisks represent that no instances solved in time.*

$\|\mathcal{P}\|$	Pure MIP	Network	iLBBD1	cLBBD1	iLBBD2p	cLBBD2p	cLBBD4p
20	0.8406	94.56	0.2942	0.6679	0.6466	0.6335	0.6047
40	5.448	17.52	$28.90^{(1)}$	58.87	$52.35^{(1)}$	$46.21^{(1)}$	$6.077^{(1)}$
60	6.012	29.11	101.8	15.74	$12.01^{(1)}$	45.06	36.22
80	9.716	30.48	34.90	$71.66^{(1)}$	18.79	62.25	175.4

Table 5: *Average gap (%) over 5 instances after trying to solve to optimality with a 1% gap. MIPGap is reported for pure MIP, Network and callback implementations of LBBD. Gap between master problem lowerbound and best sub problem upperbound is reported for iterative implementations of LBBD.*

$\|\mathcal{P}\|$	Pure MIP	Network	iLBBD1	cLBBD1	iLBBD2p	cLBBD2p	cLBBD4p
20	0.9675	0.903	0.0	0.7989	0.0	0.9242	0.9166
40	0.9005	0.5333	0.3343	0.8768	0.6572	0.8798	0.8963
60	0.7684	0.7727	0.2791	0.8142	0.7442	0.7041	0.9207
80	0.7875	0.8781	0.3236	0.983	0.4335	0.8632	0.7431

6 Discussion

It should be emphasised that because our time constraints are were stricter robustness may have been greatly affected giving different results to the original paper, however we found that our results were already disparate without this factor. For example, the original paper was able to solve with patient sizes up to 100 without ever going over an average run-time of 200 seconds.

We speculate that implementation

7 Conclusion

Discretizing time further into periods may remedy the scaling issues found with the network model, reducing the number of nodes and arcs at the cost of accuracy.

A Appendix

Table 6: *Time taken for each instance when trying to solve to optimality.*

seed	$\|\mathcal{P}\|$	Pure MIP	Network	iLBBD1	cLBBD1	iLBBD2p	cLBBD2p	cLBBD4
42	20	55.621249	900	4.123265	0.949113	3.958082	0.605675	0.654821
831	20	18.126045	900	0.296833	0.253265	0.274671	0.416003	0.441198
306	20	2.041458	900	1.361992	2.055692	1.106399	1.808341	1.716485
542	20	3.000340	900	0.580737	0.504805	0.600731	0.514555	0.520827
1	20	1.505760	900	1.181184	0.651583	1.215210	1.055650	0.611608
42	40	39.046759	64.643897	901.969151	900.928267	900.922126	902.530971	901.404782
831	40	900.074868	472.309139	1171.195916	900.966696	1605.561038	900.273473	900.329041
306	40	3.928887	900.159554	4.428572	1.911088	4.503378	1.910761	1.959761
542	40	900.199757	900.153089	1045.090099	901.370528	907.687432	901.559075	900.948533
1	40	494.895451	1394.601143	905.294060	905.809085	935.405877	900.435013	904.002700
42	60	900.136048	900	902.127126	900.441417	905.850299	903.361533	900.170979
831	60	900.159431	900	912.129558	900.112896	1183.786823	900.232333	902.206866
306	60	30.804891	900	24.459900	10.726104	34.606863	21.455653	25.795459
542	60	900.154018	900	901.279073	900.676095	919.253635	901.173662	901.073426
1	60	900.127659	900	1071.186251	900.254857	1084.595362	900.801214	900.983243
42	80	900.069266	900.105599	920.841381	901.018549	949.759938	900.152319	900.167128
831	80	900.084165	900.246634	1491.767385	900.493704	1251.183445	901.669027	901.097375
306	80	900.093483	900.246199	1307.066104	900.037347	1297.656757	900.081336	901.133256
542	80	900.088750	900.201944	903.723640	900.139623	903.938128	900.088399	903.650919
1	80	900.167469	900.200223	900.285928	901.299396	900.273110	900.055926	900.098382

Table 7: *Gap reached for each instance when trying to solve to optimality.*

seed	$\|\mathcal{P}\|$	Pure MIP	Network	iLBBD1	cLBBD1	iLBBD2p	cLBBD2p	cLBBD4
42	20	0.0	0.002519	0.0	0.0	0.0	0.0	0.0
831	20	0.0	0.000665	0.0	0.0	0.0	0.0	0.0
306	20	0.0	0.006489	0.0	0.0	0.0	0.0	0.0
542	20	0.0	0.006600	0.0	0.0	0.0	0.0	0.0
1	20	0.0	0.008476	0.0	0.0	0.0	0.0	0.0
42	40	0.000000	0.0	0.0	0.004805	0.000000	0.011210	0.011384
831	40	0.000136	0.0	0.0	0.007263	0.000476	0.007375	0.007816
306	40	0.000000	0.007059	0.0	0.000000	0.000000	0.000000	0.000000
542	40	0.001973	0.003665	0.0	0.001876	0.000000	0.001533	0.001893
1	40	0.000000	0.006188	0.0	0.003714	0.000000	0.003705	0.003645
42	60	0.003867	0.002708	0.00000	0.004018	0.000000	0.005890	0.005160
831	60	0.001023	0.002575	0.00000	0.002758	0.000024	0.002752	0.005001
306	60	0.000000	0.003802	0.00000	0.000000	0.000000	0.000000	0.000000
542	60	0.001695	0.007050	0.00000	0.003204	0.000000	0.006188	0.003469
1	60	0.002548	0.004519	0.00004	0.002722	0.000061	0.002733	0.002733
42	80	0.002371	0.000983	0.000000	0.012166	0.000000	0.002948	0.005397
831	80	0.003397	0.001125	0.000000	0.004568	0.000019	0.003980	0.007607
306	80	0.002452	0.001983	0.000045	0.004187	0.000011	0.006921	0.003248
542	80	0.000501	0.002840	0.000014	0.003439	0.000024	0.003772	0.002691
1	80	0.004427	0.003308	0.000049	0.005641	0.000050	0.005971	0.007148

Table 8: *Objective values when trying to solve to optimality.*

seed	$\|\mathcal{P}\|$	Pure MIP	Network	iLBBD1	cLBBD1	iLBBD2p	cLBBD2p	cLBBD4
42	20	-312419.0	-312419.0	-312419.0	-312419.000000	-312419.0	-312419.0	-312419.0
831	20	-187989.0	-187989.0	-187989.0	-187989.000000	-187989.0	-187989.0	-187989.0
306	20	-220079.0	-220079.0	-220079.0	-220079.000000	-220079.0	-220079.0	-220079.0
542	20	-238297.0	-238030.0	-238297.0	-238297.000000	-238297.0	-238297.0	-238297.0
1	20	-244112.0	-244112.0	-244112.0	-244112.002591	-244112.0	-244112.0	-244112.0
42	40	-503251.0	-503151.0	-503651.0	-501294.0	-503651.0	-498116.0	-498046.0
831	40	-367748.0	-367748.0	-368833.0	-366247.0	-368833.0	-366197.0	-366047.0
306	40	-383537.0	-383487.0	-383537.0	-383537.0	-383537.0	-383537.0	-383537.0
542	40	-412576.0	-413122.0	-413322.0	-412576.0	-413322.0	-412722.0	-412576.0
1	40	-402834.0	-402067.0	-403067.0	-401653.0	-403067.0	-401653.0	-401653.0
42	60	-741357.0	-744057.0	-744307.0	-741357.0	-744307.0	-740007.0	-740542.0
831	60	-654542.0	-654683.0	-655215.0	-653458.0	-655215.0	-653458.0	-652005.0
306	60	-564312.0	-564212.0	-564312.0	-564312.0	-564312.0	-564312.0	-564312.0
542	60	-675220.0	-673723.0	-675400.0	-673288.0	-675400.0	-671849.0	-673117.0
1	60	-602474.0	-602174.0	-604042.0	-602474.0	-604042.0	-602474.0	-602474.0
42	80	-957258.0	-959091.0	-959591.0	-948632.0	-959591.0	-957158.0	-955093.0
831	80	-830774.0	-832624.0	-833293.0	-829544.0	-833293.0	-830028.0	-827186.0
306	80	-801771.0	-803513.0	-803663.0	-800453.0	-803663.0	-798478.0	-801133.0
542	80	-892084.0	-892173.0	-892573.0	-889573.0	-892573.0	-889273.0	-890231.0
1	80	-806619.0	-808385.0	-809624.0	-805178.0	-809624.0	-805027.0	-804203.0

References

[1] Vahid Roshanaei et al. "Propagating logic-based Benders' decomposition approaches for distributed operating room scheduling". In: *European Journal of Operational Research* 257.2 (2017), pp. 439–455.

[2] Cheng Guo et al. "Logic-based Benders Decomposition and Binary Decision Diagram Based Approaches for Stochastic Distributed Operating Room Scheduling". In: (July 2019).

[3] John Hooker. "Logic-Based Benders Decomposition for Large-Scale Optimization". In: Sept. 2019, pp. 1–26. ISBN: 978-3-030-22787-6. DOI: 10.1007/978-3-030-22788-3_1.

[4] Mohammad M. Fazel-Zarandi and J. Christopher Beck. "Using Logic-Based Benders Decomposition to Solve the Capacity- and Distance-Constrained Plant Location Problem". In: *INFORMS Journal on Computing* 24.3 (2012), pp. 387–398. DOI: 10.1287/ijoc.1110.0458. eprint: https://doi.org/10.1287/ijoc.1110.0458. URL: https://doi.org/10.1287/ijoc.1110.0458.