Online Supplement – “A constraint-programming-based branch-and-price-and-cut approach for operating room planning and scheduling”

Seyed Hossein Hashemi Doulabi[1,3], **Louis-Martin Rousseau**[1,3], **Gilles Pesant**[2,3]

[1]Department of Mathematics and Industrial Engineering, Polytechnique Montréal, Montreal, Canada

[2]Department of Computer and Software Engineering, Polytechnique Montréal, Montreal, Canada

[3]Interuniversity Research Center on Enterprise Networks, Logistics and Transportation (CIRRELT), Montreal, Canada

Appendix 1- The model for the case of nonidentical operating rooms

In this appendix, we discuss how the CG algorithm proposed in Section 3 can take into account the case of nonidentical operating rooms. We consider two cases:

Case 1) Different availabilities: In his case, instead of considering a separate subproblem for each operating room, the presented constraint programming model can be modified by adding dummy mandatory surgeries to represent the unavailable early and late time periods. We add the following constraints to the subproblems:

$$\text{If } (W_1 = i) \text{ then } (V_1 = 0) \qquad \forall i \in dummy_{early} \qquad \text{(A1-1)}$$

$$\text{If } \left(W_p = j\right) \text{ then } \left(V_p + t_{[W_p]} = |T_d|\right) \qquad \forall j \in dummy_{late}, \forall p \in \{2, \dots, n\} \qquad \text{(A1-2)}$$

We assume that the early dummy surgeries must be scheduled in the first position and the others can be scheduled in any other positions. We specify appropriate deadlines to ensure that the dummies are assigned to the correct days. We must schedule appropriate pairs of dummy surgeries together; for this we require the following constraint:

$$\text{If } (W_1 = i) \text{ then } \left(C_{comp(i)} == 1\right) \qquad \forall i \in dummy_{early} \qquad \text{(A1-3)}$$

This constraint states that if a dummy surgery is scheduled in the morning, its complementary dummy surgery, denoted $comp(i)$, must appear in the same schedule. Constraint (A1-2) forces $comp(i)$ to be the last surgery in the operating room.

2) Different equipment: For this case we need a different subproblem for each type of operating room. In the master problem, we modify constraint (4) as follows:

$$\sum_{j \in J_d} x_j \leq \left|K_{r,d}\right| \qquad \forall d \in D, \forall r \in Room_d \qquad \text{(A1-4)}$$

where $Room_d$ is the set of operating room types on day d, and $\left|K_{r,d}\right|$ is the number of rooms of type r on day d. The model still breaks the symmetry of identical rooms. Note that rooms of the same type have the same equipment but may have different availabilities.

Appendix 2- An integer programming model adapted from (Marques et al. 2012)

The problem defined in Section 2 can be formulated as an integer programming model using the four-index binary variables x_{ikdt} (Roland et al. 2006, Roland et al. 2010, Marques et al. 2012) and two other sets of binary variables y_{ikd} and $z_{ii'kd}$. We define the following sets and variables.

Sets:

T_{di}	:	A subset of T_d such that if surgery i starts at any time slot in this subset it finishes before the end of regular time on day d.
$Pairs_d$	:	1) The set of all pairs of surgeries i and i' with deadlines on or after day d such that cleaning is required if surgery i' is performed immediately after surgery i.

Variables:

x_{ikdt}	:	1 if surgery i starts at time t on day d in operating room k; 0 otherwise.
y_{ikd}	:	1 if surgery i is scheduled in operating room k on day d; 0 otherwise.
$z_{ii'kd}$	:	1 if the pair $(i, i') \in Pairs_d$ is scheduled in operating room k on day d and surgery i starts at any time before surgery i' (not necessarily successively).

The integer programming model is as follows:

$$Max \sum_{d\in D} \sum_{k\in K_d} \sum_{i\in I:d_i\geq d} \sum_{t'\in T_{di}} (t_i x_{ikdt'}) \qquad \text{(A2-1)}$$

subject to:

$$\sum_{d\in D:d\leq d_i} \sum_{k\in K_d} \sum_{t\in T_{di}} x_{ikdt} = 1 \qquad \forall i \in I_1 \qquad \text{(A2-2)}$$

$$\sum_{d\in D:d\leq d_i} \sum_{k\in K_d} \sum_{t\in T_{di}} x_{ikdt} \leq 1 \qquad \forall i \in I_2 \qquad \text{(A2-3)}$$

$$\sum_{i\in I:d_i\geq d} \sum_{t'\in T_{di}:max[t-t_i,0]<t'\leq t} x_{ikdt'} \leq 1 \qquad \forall d \in D, \forall k \in K_d, \forall t \in T_d \qquad \text{(A2-4)}$$

$$\sum_{i\in I'_l:d_i\geq d} \sum_{k\in K_d} \sum_{t'\in T_{di}:max[t-t_i,0]<t'\leq t} x_{ikdt'} \leq 1 \qquad \forall d \in D, \forall l \in L, \forall t \in T_d \qquad \text{(A2-5)}$$

$$\sum_{i\in I'_l:d_i\geq d} \sum_{k\in K_d} \sum_{t'\in T_{di}} t_i x_{ikdt'} \leq A_{ld} \qquad \forall l \in L, \forall d \in D \qquad \text{(A2-6)}$$

$$y_{ikd} = \sum_{t\in T_{di}} x_{ikdt} \qquad \forall d \in D, \forall k \in K_d, \forall i \in I: d_i \geq d \qquad \text{(A2-7)}$$

$$z_{ii'kd} \leq y_{ikd},\ z_{i'ikd} \leq y_{i'kd} \qquad \forall d \in D, \forall k \in K_d, \forall (i, i') \in Pairs_d \qquad \text{(A2-8)}$$

$$z_{ii'kd} + z_{i'ikd} \leq 1 \qquad \forall d \in D, \forall k \in K_d, \forall (i, i') \in Pairs_d \ (i < i') \qquad \text{(A2-9)}$$

$$z_{ii'kd} + z_{i'ikd} \geq y_{ikd} + y_{i'kd} - 1 \qquad \forall d \in D, \forall k \in K_d, \forall (i, i') \in Pairs_d \ (i < i') \qquad \text{(A2-10)}$$

$$\sum_{t\in T_{di}} (t x_{ikdt}) + t_i + CL_{ii'} \leq \sum_{t\in T_{di'}} (t x_{i'kdt}) + M(1 - z_{ii'kd}) \qquad \forall d \in D, \forall k \in K_d, \forall (i, i') \in Pairs_d \qquad \text{(A2-11)}$$

$$x_{ikdt} \in \{0,1\} \qquad \forall d \in D, \forall k \in K_d, \forall t \in T_d, \forall i \in I: d_i \geq d \qquad \text{(A2-12)}$$

$$z_{ii'kd} \in \{0,1\} \qquad \forall d \in D, \forall k \in K_d, \forall (i, i') \in Pairs_d \qquad \text{(A2-13)}$$

The objective function (A2-1) maximizes the total duration of the scheduled surgeries. Constraints (A2-2) and (A2-3) ensure that the mandatory surgeries are performed and allow the optional surgeries to be postponed. Constraint (A2-4) ensures that each operating room on each day is occupied by at most one surgery in each time slot. Constraint (A2-5) prevents overlapping surgeries for the same surgeon. Constraint (A2-6) enforces the maximum working hours for each surgeon. Constraint (A2-7) links the intermediate variables y_{ikd} to x_{ikdt}; these variables are included to make the model more readable. Constraints (A2-8) force $z_{ii'kd}$ to be 0 if surgery i or surgery i' is not scheduled in operating room k on day d. Constraints (A2-9) and (A2-10) together force either $z_{ii'kd}$ or $z_{i'ikd}$ to be 1 if surgeries i and i' are both scheduled in operating room k on day d. If two surgeries from the set $Pairs_d$ are scheduled in the same room on the same day, they must be sequenced. Constraint (A2-11) ensures that if $z_{ii'kd}$ is 1, the start time of surgery i' occurs after the finish time of surgery i in room k on day d plus $CL_{ii'}$.

Appendix 3- A pure constraint-programming model

We now present a constraint programming model for the problem defined in Section 2. The constraint programming variables are as follows:

y_{id} : It is an interval variable for surgery i on day d. This variable is present in the solution if surgery i is scheduled on day d.

z_{id} : This variable is the index of the surgery scheduled immediately before surgery i in the same operating room on day d. If surgery i is the first surgery in the room this variable is 0 and if surgery i is not scheduled on day d it is -1.

The following model is implemented using the CP optimizer in IBM ILOG CPLEX Optimization Studio V12.4.

$$Max\{\sum_{d\in D}\sum_{i\in I:dd_i\geq d} Length(y_{id})\} \qquad \text{(A3-1)}$$

subject to:

$$z_{id} \in \{i' \in I \mid i' \neq i, dd_{i'} \geq d\} \cup \{0,-1\} \qquad \text{(A3-2)}$$

$$End(y_{id}) \leq |T_d| \qquad \forall d \in D, \forall i \in I: dd_i \geq d \qquad \text{(A3-3)}$$

$$\text{Count}([z_{id}]_{i\in I:dd_i\geq d}, 0) \leq |K_d| \qquad \forall d \in D \qquad \text{(A3-4)}$$

$$\text{Count}\left([z_{id}]_{i\in I\backslash\{i'\}:dd_i\geq d}, i'\right) \leq Presence(y_{i'd}) \qquad \forall d \in D, \forall i' \in I: dd_{i'} \geq d \qquad \text{(A3-5)}$$

$$\text{NoOverlap}([y_{id}]_{i\in I'_l:dd_i\geq d}) \qquad \forall d \in D, \forall l \in L \qquad \text{(A3-6)}$$

$$\begin{array}{ll} \text{If } (z_{id} = i') \text{ then} & \forall d \in D, \forall i \in I: dd_i \geq d, \\ (Presence(y_{i'd}) = 1) \wedge (Start(y_{id}) \geq End(y_{i'd}) + CL_{i'i}) & \forall i' \in I\backslash\{i\}: dd_{i'} \geq d \end{array} \qquad \text{(A3-7)}$$

$$\text{If } (z_{id} = -1) \text{ then } (Presence(y_{id}) = 0) \qquad \forall d \in D, \forall i \in I: dd_i \geq d \qquad \text{(A3-8)}$$

$$\sum_{d\in D:dd_i\geq d} Presence(y_{id}) = 1 \qquad \forall i \in I_1 \qquad \text{(A3-9)}$$

$$\sum_{d\in D:dd_i\geq d} Presence(y_{id}) \leq 1 \qquad \forall i \in I_2 \qquad \text{(A3-10)}$$

$$\sum_{i\in I'_l:dd_i\geq d} Length(y_{id}) \leq A_{ld} \qquad \forall d \in D, \forall l \in L \qquad \text{(A3-11)}$$

The objective function (A3-1) maximizes the total scheduled surgery time. Here $Length(y_{id})$ is equal to the duration of surgery i if variable y_{id} is present in the solution. (A3-2) specifies the domains of the z_{id} variables. Constraint (A3-3) ensures that if surgery i is scheduled on day d it is completed before the end of the day. In constraint (A3-4), $\text{Count}([z_{id}]_{i\in I:dd_i\geq d}, 0)$ counts the number of times that the value 0 appears in the vector $[z_{id}]_{i\in I:dd_i\geq d}$. This constraint ensures that the number of surgeries that are the first surgeries in their operating rooms on day d does not exceed the number of available operating rooms on this day. The left-hand side of constraint (A3-5) counts the number of times that surgery i' is the immediate predecessor of the other surgeries on day d. Constraint (A3-5) indicates that if surgery i' is not scheduled on day d, it cannot be the immediate predecessor of any surgery on that day. Constraint (A3-6) prevents overlapping surgeries for the same surgeon. Constraint (A3-7) states that if surgery i' is the immediate predecessor of surgery i on day d then surgery i' must be scheduled on the same day and the start time of surgery i must be after the finish time of surgery i' plus $CL_{i'i}$. Constraint (A3-8) ensures that surgery i is not scheduled on day d if z_{id} is -1. Constraint (A3-9) ensures that the mandatory surgeries are performed, and constraint (A3-10) allows the optional surgeries to be postponed. Constraint (A3-11) enforces the maximum working hours for each surgeon.

We apply dominance rule 1 by adding the following constraint:

$$Start(y_{id}) = 0 \text{ or } Start(y_{id}) \geq SD_d \qquad \forall d \in D, \forall i \in I: dd_i \geq d$$

where as defined in Dominance rule 1, SD_d is the smallest surgery duration among all surgeries with deadlines on or after day d. (i.e. $SD_d = min_{i \in I;\, dd_i \geq d}\{t_i\}$)

We order the y_{id} variables by surgery deadline. For a fixed $i \in I$, they are ordered lexicographically by d. Each z_{id} variable is evaluated immediately after its corresponding y_{id}.

Appendix 4- Proof of λ values for different cases

We now validate the λ values in Table 1. Let Δ_1 be the cleaning time required between the surgeries in positions $p^* - 1$ and p^* if surgery i in position p^* is replaced by surgery i'. Similarly, let Δ_2 be the cleaning time required between the surgeries in positions p^* and $p^* + 1$ if surgery i in position p^* is replaced by surgery i'. We show that regardless of the infection status of the surgery in position $p^* + 1$, $\Delta_1 + \Delta_2 \leq \lambda$ holds for the λ values in Table 1. Let f'_{p^*+1} be the infection status of the surgery in position $p^* + 1$ if it is infectious. In tables presented in this appendix, different infection statuses for the surgery in position $p^* + 1$ are presented under Column "$p^* + 1$".

Case 1: Since surgeries i and i' are both noninfectious, replacing surgery i by surgery i' does not change the possible pre/post-cleaning time: $\lambda = 0$.

Case 2: Two subcases are possible.

Case 2-1- The surgery in position $p^* - 1$ is noninfectious. Table A4-1 computes Δ_1 and Δ_2 given the infection status of the surgery in position $p^* + 1$. We have $\Delta_1 + \Delta_2 \leq OCT$ and $\lambda = OCT$.

Table A4-1- Validity of λ for case 2-1.

$p^* + 1$	Δ_1	Δ_2	$\Delta_1 + \Delta_2$
Noninfectious	0	OCT	OCT
Infectious with $f'_{p^*+1} = f_{i'}$	0	0	0
Infectious with $f'_{p^*+1} \neq f_{i'}$	0	OCT	OCT
$\lambda = Max(\Delta_1 + \Delta_2)$			OCT

Case 2-2- The surgery in position $p^* - 1$ is infectious with $f'_{p^*+1} \neq f_{i'}$. Table A4-2 shows that $\lambda = OCT$.

Table A4-2- Validity of λ for case 2-2.

$p^* + 1$	Δ_1	Δ_2	$\Delta_1 + \Delta_2$
Noninfectious	0	OCT	OCT
Infected with $f'_{p^*+1} = f_i$	0	0	0
Infected with $f'_{p^*+1} \neq f_i$ and $f'_{p^*+1} \neq f_{i'}$	0	OCT	OCT
$\lambda = Max(\Delta_1 + \Delta_2)$			OCT

Case 3: Table A4-3 shows that $\lambda = 0$.

Table A2-3- Validity of λ for case 3.

$p^* + 1$	Δ_1	Δ_2	$\Delta_1 + \Delta_2$
Noninfectious	$-OCT$	OCT	0
Infectious with $f'_{p^*+1} = f_{i'}$	$-OCT$	0	$-OCT$
Infectious with $f'_{p^*+1} \neq f_{i'}$	$-OCT$	OCT	0
$\lambda = Max(\Delta_1 + \Delta_2)$			0

Case 4: Table A4-4 shows that $\lambda = 0$.

Table A4-4- Validity of λ for case 4.

$p^* + 1$	Δ_1	Δ_2	$\Delta_1 + \Delta_2$
Noninfectious	0	$-OCT$	$-OCT$
Infectious with $f'_{p^*+1} = f_i$	0	0	0
Infectious with $f'_{p^*+1} \neq f_i$	0	$-OCT$	$-OCT$
$\lambda = Max(\Delta_1 + \Delta_2)$			0

Case 5: Table A4-5 shows that $\lambda = OCT$.

Table A4-5- Validity of λ for case 5.

$p^* + 1$	Δ_1	Δ_2	$\Delta_1 + \Delta_2$
Noninfectious	OCT	$-OCT$	0
Infectious with $f'_{p^*+1} = f_i$	OCT	0	OCT
Infectious with $f'_{p^*+1} \neq f_i$	OCT	$-OCT$	0
$\lambda = Max(\Delta_1 + \Delta_2)$			OCT

Case 6: Since surgeries i and i' have the same type of infection, replacing surgery i by surgery i' does not change the possible pre/post-cleaning time: $\lambda = 0$.

Case 7: Table A4-6 shows that $\lambda = OCT$.

Table A4-6- Validity of λ for case 7.

$p^* + 1$	Δ_1	Δ_2	$\Delta_1 + \Delta_2$
Noninfectious	0	0	0
Infectious with $f'_{p^*+1} = f_i$	0	OCT	OCT
Infectious with $f'_{p^*+1} = f_{i'}$	0	$-OCT$	$-OCT$
Infectious with $f'_{p^*+1} \neq f_i$ and $f'_{p^*+1} \neq f_{i'}$	0	0	0
$\lambda = Max(\Delta_1 + \Delta_2)$			OCT

Case 8: Table A4-7 shows that $\lambda = 2OCT$.

Table A4-7- Validity of λ for case 8.

$p^* + 1$	Δ_1	Δ_2	$\Delta_1 + \Delta_2$
Noninfectious	OCT	0	OCT
Infectious with $f'_{p^*+1} = f_i$	OCT	OCT	$2OCT$
Infectious with $f'_{p^*+1} = f_{i'}$	OCT	$-OCT$	0
Infectious with $f'_{p^*+1} \neq f_i$ and $f'_{p^*+1} \neq f_{i'}$	OCT	0	OCT
$\lambda = Max(\Delta_1 + \Delta_2)$			$2OCT$

Case 9: Table A4-8 shows that $\lambda = 0$.

Table A4-8- Validity of λ for case 9.

$p^* + 1$	Δ_1	Δ_2	$\Delta_1 + \Delta_2$
Noninfectious	$-OCT$	0	$-OCT$
Infectious with $f'_{p^*+1} = f_i$	$-OCT$	OCT	0
Infectious with $f'_{p^*+1} = f_{i'}$	$-OCT$	$-OCT$	$-2OCT$
Infectious with $f'_{p^*+1} \neq f_i$ and $f'_{p^*+1} \neq f_{i'}$	$-OCT$	0	$-OCT$
$\lambda = Max(\Delta_1 + \Delta_2)$			0

Appendix 5- Comparison of branches (1) and (2) with branch (3)

Table A5-1 gives results for the three branching rules presented in Section 5-2. We compare two branch-and-price-and-cut algorithms: in Algorithm 1 we use branches 1 and 2, and in Algorithm 2 we use branch 3. The nodes are pruned whenever no branching is possible. Algorithm 1 may terminate early because no branch can be found. For a fair comparison, we first run Algorithm 1 and we then run Algorithm 2 for the same period of time; this value is given in the *Time* column. The rightmost columns give the differences in the upper and lower bounds found by the two algorithms. These values indicate that Algorithm 1 generally provides better upper bounds. Algorithm 2 generally provides better lower bounds, mainly because it generates more columns.

Table A5-1- Evaluation of the branching procedures.

# surgeries	Time	Algorithm 1				Algorithm 2				$UB_2 - UB_1$	$LB_2 - LB_1$
		# columns	UB	LB	Gap (%)	# columns	UB	LB	Gap (%)		
40	41	485	1403	1402	0.07	855	1411	1403	0.57	8	1
60	181	1214	1971	1910	3.09	1973	1975	1958	0.86	4	48
80	1341	2384	2203	2154	2.01	3391	2211	2170	1.61	8	16
100	6030	3871	2297	2247	2.14	5491	2297	2251	1.99	0	4
120	3231	3343	2304	2252	2.26	4123	2306	2254	2.24	2	2
40	147	626	1362	1361	0.06	838	1367	1359	0.61	5	-2
60	1054	1629	1854	1834	1.09	1989	1859	1839	1.09	5	5
80	2683	2497	2162	2119	2.01	3270	2173	2124	2.27	11	5
100	9049	3799	2403	2327	3.12	5477	2403	2330	3.01	0	3
120	8902	4407	2457	2342	4.67	5562	2457	2343	4.61	0	1
60	7379	3468	576	561	2.50	3472	576	571	0.66	0	10
80	7429	1889	792	702	10.68	4216	792	713	9.34	0	11
100	6823	3593	978	853	12.24	4271	978	860	11.51	0	7
					3.53				3.11	3.31	8.54

Appendix 6- A heuristic for setting the number of operating rooms

We now present the heuristic that determines the number of operating rooms for instance sets B and C.

Step 1: For each surgeon, we divide the sum of all the surgery durations corresponding to that surgeon over the five days of the week proportional to the surgeon's maximum daily hours (being careful not to exceed the maximum value). This gives an indication of the surgeon's daily working time. This might give the result presented in Table A6-1 for an example with eight surgeons.

Table A6-1- The output of Step 1 in the heuristic algorithm proposed for setting the number of operating rooms

Surgeon / **Day**	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**
Mon.	50.8	90.9	40.7	73.4	29.6	92.1	120.0	80.5
Tue.	93.8	68.4	72.8	60.5	48.4	69.1	36.6	20.3
Wed.	49.3	49.2	18.9	58.8	83.0	37.5	24.1	52.5
Thu.	58.5	24.4	112.8	59.2	63.2	30.0	72.5	69.4
Fri.	23.1	34.3	25.7	22.5	37.8	34.4	43.4	70.6

Step 2: For each day, we sum the expected daily hours to determine the total operating time. For the example, this gives the result presented in Table A6-2.

Table A6-2- The output of Step 2 in the heuristic algorithm proposed for setting the number of operating rooms

Day	Mon.	Tue.	Wed.	Thu.	Fri.
Total operating time	578.0	469.8	373.4	490.1	291.8

Step 3: We divide the total times by eight hours (i.e., 96 five-minute intervals) to determine the expected number of operating rooms for each day.

Table A6-3- The output of Step 3 in the heuristic algorithm proposed for setting the number of operating rooms

Day	Mon.	Tue.	Wed.	Thu.	Fri.
Number of operating rooms	6	5	4	5	3