RELATÓRIO SPRINT-2

ALGAV

Instituto Superior de Engenharia do Porto

2024/2025

3DG – Grupo 38

1221933 | Ana Sofia De Oliveira Guterres

1220738 | José Afonso Calheiros Cruz Dos Santos

1220683 | Matilde Xavier Varela

1220841 | Rita Beatriz Ferreira Barbosa

ISEP INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

ÍNDICE

EXPLICAÇÃO CÓDIGO BASE

Para o desenvolvimento da solução requisitada, utilizámos uma série de predicados base com o objetivo de agendar as cirurgias considerando apenas os doutores.

O predicado ***obtain_better_sol/5*** é responsável por realizar o agendamento das cirurgias utilizando uma abordagem de força bruta. Esta estratégia gera todas as possibilidades de agendamento, com o objetivo de encontrar a solução ótima, definida como o menor tempo possível para a realização da última cirurgia.

Para alcançar este objetivo, o predicado ***availability_all_surgeries/3*** é invocado a partir de ***obtain_better_sol1/3***. Este último utiliza o predicado ***permutation/2*** para gerar todas as combinações possíveis de agendamento. Em seguida, as combinações são avaliadas, cirurgia a cirurgia, através do predicado ***availability_operation/5***.

O predicado ***availability_operation/5*** é responsável por verificar os intervalos de tempo disponíveis para a realização de cada cirurgia. Para isso, utiliza os predicados ***intersect_all_agendas/3*** e ***intersect_2_agendas/3***, que determinam os intervalos de tempo comuns em que tanto os doutores como a sala de operações estão simultaneamente disponíveis.

Por fim, o predicado ***update_better_sol/4*** é utilizado dentro de ***obtain_better_sol1/3*** para atualizar a melhor solução. Para cada combinação gerada em *obtain_better_sol1/3* e processada pelo predicado ***availability_all_surgeries/3***, ***update_better_sol/4*** avalia se a combinação em análise representa uma solução melhor do que a solução anterior armazenada. Caso a combinação seja mais eficiente (ou seja, tenha um menor tempo final para a última cirurgia), o predicado atualiza a solução em ***better_sol/5***.

Adicionalmente, é importante destacar a existência de predicados auxiliares que desempenham um papel fundamental no desenvolvimento da solução. Entre eles, destacam-se ***adapt_timetable/4*** e ***free_agenda0/2***.

O predicado ***adapt_timetable/4*** ajusta os tempos livres dos doutores, assegurando que respeitam os respetivos horários de trabalho. Por outro lado, ***free_agenda0/2*** é responsável por obter os intervalos de tempo livres, tendo em conta os períodos já ocupados nas agendas.

Estes predicados garantem uma gestão eficiente e precisa dos horários disponíveis para o agendamento das cirurgias.

ESTUDO DA COMPLEXIDADE

Resultados obtidos para o algoritmo da melhor solução

Nº de cirurgias	Nº de soluções	Melhor agendamento das atividades na sala de operações	Tempo final da última cirurgia (minutes)	Tempo demorado para obter soluções (s)
3	6	[(480, 539, so100001), (540, 629, so100002), (630, 704, so100003), (750, 780, mnt0001), (1080, 1110, mnt0002)]	704	0,0464
4	24	[(520, 594, so100003), (595, 654, so100004), (655, 744, so100002), (750, 780, mnt0001), (791, 850, so100001), (1080, 1110, mnt0002)]	850	0,1642
5	120	[(500, 559, so100004), (560, 634, so100005), (635, 724, so100002), (750, 780, mnt0001), (791, 865, so100003), (866, 925, so100001), (1080, 1110, mnt0002)]	925	0,8129
6	720	[(500,559,so100004),(560,634,so100005),(635,724,so100002),(750,780,mnt0001),(791,865,so100003),(866,925,so100001),(926,985,so100006),(1080,1110,mnt0002)]	985	4,677
7	5040	[(500,559,so100004),(560,649,so100007),(650,739,so100002),(750,780,mnt0001),(791,865,so100003),(866,925,so100001),(926,985,so100006),(986,1060,so100005),(1080,1110,mnt0002)]	1060	26,22
8	40320	[(500,589,so100002),(590,649,so100004),(650,709,so100008),(750,780,mnt0001),(791,865,so100003),(866,925,so100001),(926,985,so100006),(986,1075,so100007),(1080,1110,mnt0002),(1111,1185,so100005)]	1185	39,31
9	362880	[(480,539,so100001),(540,629,so100002),(630,689,so100008),(690,749,so100009),(750,780,mnt0001),(791,880,so100007),(881,940,so100006),(961,1020,so100004),(1080,1110,mnt0002),(1111,1185,so100005),(1186,1260,so100003)]	1260	222,8
10	3628800	[(500,559,so100004),(560,619,so100010),(620,679,so100008),(680,739,so100009),(750,780,mnt0001),(791,865,so100003),(866,925,so100001),(926,985,so100006),(986,1075,so100002),(1080,1110,mnt0002),(1111,1185,so100005),(1186,1275,so100007)]	1275	732,9
11	39916800	[(480,539,so100001),(540,629,so100002),(630,689,so100008),(690,749,so100009),(750,780,mnt0001),(791,880,so100007),(881,940,so100006),(941,1015,so100011),(1016,1075,so100004),(1080,1110,mnt0002),(1111,1185,so100003),(1186,1260,so100005),(1261,1320,so100010)]	1320	4614

Como se pode observar, entre 2 e 6 cirurgias, o algoritmo apresenta resultados de forma bastante eficiente, com tempos de execução reduzidos. No entanto, a partir de 6 cirurgias, o tempo de execução começa a aumentar ligeiramente até às 8 cirurgias. A partir deste ponto, o aumento do tempo de execução torna-se significativo. Este aumento

torna-se mais acentuado ao atingir as 11 cirurgias, onde o tempo de processamento chega a 4614 segundos, ou seja, 1 hora e 16 minutos.

Posto isto, apesar de apresentar ótimos resultados, este algoritmo é ineficiente para o agendamento de várias cirurgias (a partir de 8 cirurgias), uma vez que não é capaz de apresentar resultados em tempo real.

HEURÍSTICAS

Critério 1 – Doutor com disponibilidade mais cedo

O objetivo desta solução é agendar as cirurgias de acordo com o doutor que se encontra disponível para fazer as cirurgias lhe atribuídas completamente. Ou seja, é necessário saber todos os doutores associados às cirurgias que pretendemos agendar, e posto isto, encontrar aquele que pode a fazer mais cedo.

Tendo isso em conta, desenvolvemos o predicado ***obtain_heuristic_solution(Room, Day, AgOpRoomHeuristic, LAgDoctorsHeuristic, TFinOp)***. Este predicado recorre a ***obtain_heuristic_solution1(Room, Day)***, que obtém os códigos das cirurgias a serem agendadas, prepara os predicados auxiliares necessários para a execução da solução e, em seguida, chama o predicado ***availability_early_surgeries(LOC, Room, Day)*** para agendar as cirurgias.

O propósito do predicado ***availability_early_surgeries(LOC, Room, Day)*** é efetuar o agendamento das cirurgias, criando as agendas para os doutores e as salas de operações, priorizando o doutor que está disponível para realizar a cirurgia mais cedo.

Assim sendo, utilizamos o predicado ***find_earliest_surgery([OpCode|LOpCode], Room, Day)*** para obter a cirurgia com tempo de início mais cedo possível entre as cirurgias que pretendemos agendar. Para obter os horários disponíveis para uma cirurgia específica, recorremos ao **predicado *availability_operation(OpCode, Room, Day, LPossibilities, LDoctors)***, que devolve as possibilidades de agendamento para a cirurgia especificada. De seguida, verificamos se o horário de início desta cirurgia é o mais cedo entre todas. Caso seja, atualizamos o predicado auxiliar ***earliest_surgery(OpCode, TinS, LDoctors)*** com os dados da cirurgia correspondente.

Quando o predicado ***find_earliest_surgery/3*** termina, teremos encontrado a cirurgia que pode ser realizada mais cedo. Esta será então agendada, e o código associado a essa cirurgia será removido da lista de operações que pretendemos agendar. Este processo repete-se até que a lista de cirurgias a agendar fique vazia.

Resultados obtidos

Ao analisar o tempo necessário para obter as soluções, verificámos que esta abordagem gera resultados significativamente mais rápidos do que a solução ótima, o que era esperado. No entanto, constatámos também que o tempo final da última cirurgia agendada é consideravelmente superior ao dos tempos obtidos na melhor solução, o que não é ideal.

Além disso, notámos que, quando o número de cirurgias a agendar ultrapassa as 8, já não é possível acomodar todas as cirurgias no mesmo dia.

Nº de cirurgias	Nº de soluções	Agendamento das atividades na sala de operações	Tempo final da última cirurgia (minutes)	Tempo demorado para obter soluções (s)
3	6	[(480, 539, so100001), (540, 629, so100002), (630, 704, so100003), (750, 780, mnt0001), (1080, 1110, mnt0002)]	704	0,0021
4	24	[(480, 539, so100001), (540, 629, so100002), (630, 704, so100003), (750, 780, mnt0001), (961, 1020, so100004), (1080, 1110, mnt0002)]	1020	0,0028
5	120	[(480, 539, so100001), (540, 629, so100002), (630, 704, so100003), (750, 780, mnt0001), (961, 1020, so100004), (1080, 1110, mnt0002), (1111, 1185, so100005)]	1185	0,0028
6	720	[(480,539,so100001),(540,629,so100002),(630,704,so100003),(750,780,mnt0001),(791,850,so100006),(961,1020,so100004),(1080,1110,mnt0002),(1111,1185,so100005)]	1185	0,0020
7	5040	[(480,539,so100001),(540,629,so100002),(630,704,so100003),(750,780,mnt0001),(791,850,so100006),(961,1020,so100004),(1080,1110,mnt0002),(1111,1185,so100005),(1186,1275,so100007)]	1275	0,0023
8	40320	[(480,539,so100001),(540,629,so100002),(630,704,so100003),(750,780,mnt0001),(791,850,so100006),(961,1020,so100004),(1080,1110,mnt0002),(1111,1185,so100005),(1186,1275,so100007)]	1275	0,0061
9	362880	[(480,539,so100001),(540,629,so100002),(630,704,so100003),(750,780,mnt0001),(781,840,so100009),(841,900,so100006),(961,1020,so100004),(1080,1110,mnt0002),(1111,1185,so100005),(1186,1275,so100007)]	1275	0,0171
10	3628800	[(480,539,so100001),(540,629,so100002),(630,704,so100003),(750,780,mnt0001),(781,840,so100009),(841,900,so100006),(961,1020,so100004),(1080,1110,mnt0002),(1111,1185,so100005),(1186,1275,so100007)]	1275	0,0320
11	39916800	[(480,539,so100001),(540,629,so100002),(630,704,so100003),(750,780,mnt0001),(781,840,so100009),(841,900,so100006),(901,975,so100011),(976,1035,so100004),(1080,1110,mnt0002),(1111,1185,so100005),(1186,1275,so100007)]	1275	0,0116

Critério 2 – Doutor com maior ocupação

O objetivo desta solução é agendar as cirurgias com base na ocupação dos doutores, ou seja, priorizar o agendamento das cirurgias do doutor que tiver a maior ocupação. A ocupação de cada doutor é calculada através da percentagem entre o tempo total das cirurgias a ele atribuídas e o tempo livre que este possui.

Assim como no critério anterior, desenvolvemos os predicados ***obtain_heuristic_highest_occupancy_solution(Room, Day, AgOpRoomBetter,***

LAgDoctorsBetter,TFinOp) e o***btain_heuristic_highest_occupancy_solution1(Room, Day)***, que seguem a mesma lógica dos predicados do critério anterior. No entanto, estes utilizam o predicado ***find_surgery_by_highest_occupancy(LOpCodes, Room, Day)***para agendar as cirurgias.

O ***find_surgery_by_highest_occupancy(LOpCodes, Room, Day)*** começa por obter uma lista de todos os doutores que estão associados às cirurgias, através do ***obtain_assignment_surgeries(LOpCodes,LDoctorsInvolved),*** e, de seguida, utiliza o predicado ***calculate_doctor_highest_occupancy_percentage(Day, LDoctorsInvolved, DocHighestPerc)*** para obter o doutor que apresenta a maior ocupação. De seguida, vai obter as cirurgias associadas a esse doutor, escolhe uma dessas e através do predicado ***availability_operation(OpCode, Room, Day, LPossibilities, LDoctors)*** obtém as possibilidades de agendamento para essa cirurgia, caso essa lista não seja vazia prossegue ao agendamento do primeiro intervalo possível e atualiza as agendas tanto dos doutores como da sala. Por fim, remove a cirurgia agendada da lista de cirurgias que pretendemos agendar (*LOpCodes*).

O processo de agendamento descrito vai ser repetido até que a lista de cirurgias esteja vazia.

O predicado ***find_surgery_by_highest_occupancy(LOpCodes, Room, Day)*** começa por obter uma lista com os doutores associados às cirurgias que pretendemos agendar, através do predicado ***obtain_assignment_surgeries(LOpCodes, LDoctorsInvolved)***. Em seguida, utiliza o predicado ***calculate_doctor_highest_occupancy_percentage(Day, LDoctorsInvolved, DocHighestPerc)*** para obter o doutor com a maior ocupação.

Após determinar o doutor com maior ocupação, o predicado obtém as cirurgias associadas a esse doutor, seleciona uma delas e, através de ***availability_operation(OpCode, Room, Day, LPossibilities, LDoctors)***, obtém as possibilidades de agendamento para a cirurgia. Caso existam intervalos disponíveis, o agendamento é feito para o primeiro horário possível, e as agendas dos doutores e da sala de operações são atualizadas. Por fim, a cirurgia agendada é removida da lista de cirurgias pendentes (*LOpCodes*).

Este processo de agendamento é repetido até que a lista de cirurgias a agendar esteja vazia.

Resultados obtidos

Tal como no critério anterior, esta abordagem gera resultados significativamente mais rápidos do que a solução ótima, mas os tempos finais são mais longos. Em adição, verificámos que esta solução apresenta tempos finais parecidos aos do critério anterior, assim como obtém os resultados em tempos parecidos.

Outro ponto a considerar é que, quando o número de cirurgias a agendar ultrapassa as 7, já não é possível acomodar todas as cirurgias no mesmo dia.

Nº de cirurgias	Nº de soluções	Agendamento das atividades na sala de operações	Tempo final da última cirurgia (minutes)	Tempo demorado para obter soluções (s)
3	6	[(520, 594, so100003), (595, 684, so100002), (750, 780, mnt0001), (791, 850, so100001), (1080, 1110, mnt0002)]	850	0,0114
4	24	[(520,594,so100003),(595,684,so100002),(750,780,mnt0001),(791,850,so100001),(961,1020,so100004),(1080,1110,mnt0002)]	1020	0,0046
5	120	[(520,594,so100003),(595,669,so100005),(750,780,mnt0001),(791,850,so100001),(961,1050,so100002),(1080,1110,mnt0002),(1141,1200,so100004)]	1200	0,0070
6	720	[(520,594,so100003),(595,669,so100005),(750,780,mnt0001),(791,850,so100001),(851,910,so100006),(961,1050,so100002),(1080,1110,mnt0002),(1141,1200,so100004)]	1200	0,0093
7	5040	[(520,594,so100003),(595,669,so100005),(750,780,mnt0001),(791,880,so100007),(881,940,so100001),(961,1050,so100002),(1080,1110,mnt0002),(1141,1200,so100004)]	1200	0,0158
8	40320	[(520,594,so100003),(620,679,so100008),(750,780,mnt0001),(791,880,so100007),(881,940,so100001),(981,1055,so100005),(1080,1110,mnt0002),(1111,1200,so100002)]	1200	0,0161
9	362880	[(520,594,so100003),(620,679,so100008),(680,739,so100009),(750,780,mnt0001),(791,880,so100007),(881,940,so100001),(981,1055,so100005),(1080,1110,mnt0002),(1111,1200,so100002)]	1200	0,0173
10	3628800	[(520,594,so100003),(620,679,so100008),(680,739,so100009),(750,780,mnt0001),(791,880,so100007),(881,940,so100001),(981,1055,so100005),(1080,1110,mnt0002),(1111,1170,so100010),(1171,1260,so100002)]	1260	0,0170
11	39916800	[(520,594,so100003),(620,679,so100008),(680,739,so100009),(750,780,mnt0001),(791,880,so100007),(881,940,so100001),(981,1055,so100005),(1080,1110,mnt0002),(1111,1170,so100010),(1171,1260,so100002)]	1260	0,0209

ALTERAÇÕES PARA INCLUIR TODO O STAFF E FASES

Relativamente a este objetivo no qual nos era pedido a mudança dos predicados base de forma a incluir os tempos de *staff* e todas as fases, tendo em consideração os seguintes aspetos:

- A equipa de anestesia precisa de apenas estar durante o tempo de preparação do paciente e o tempo da operação em si, ou seja, *TAnesthesia* + *TSurgery*
- A equipa de cirurgia precisa de apenas estar durante o tempo da operação (*TSurgery*).
- A equipa de limpeza precisa de estar presente durante o tempo da limpeza da sala (*TCleaning*).
- A sala precisa de um intervalo de tempo suficiente para realizar todas as fases, isto é, *TAnesthesia* + *TSurgery* + *TCleaning*.

O predicado base ***availability_operation*/5** foi alterado, passando agora a chamar-se ***availability_operation_changed2***, de forma a agora receber o *TAnesthesia* (tempo de anestesia) e o *TCleaning* (tempo de limpeza) da cirurgia que trata, não só isso, mas também foi criado um sistema de discernimento do staff de cada equipa, neste caso procurando o staff na sua totalidade e utilizando ***include/3*** de forma a encontrar os staffs de cada tipo nessa lista e separando-os nas suas próprias listas.

Depois é feita a interseção dos horários de todos os staffs de cada equipa, o cálculo do horário livre da sala e é calculado todos os intervalos de tempo possíveis que têm tamanho igual ao tempo total da cirurgia.

O cálculo dos intervalos em que a cirurgia é possível funciona da seguinte forma, os intervalos de tempo de tamanho igual ao tempo total da cirurgia que foram calculados anteriormente passam pelos predicados ***filter_by_availability/4***, ***filter_by_surgery_availability/5*** e ***filter_by_cleaning_availability/6*** cada um destes testa cada intervalo ao verificar se o tempo de início do trabalho da equipa em questão e o tempo final hipotético encontram-se dentro de um dos horários livres dessa equipa, se sim, é adicionado á lista de intervalos passados.

Finalmente são passadas essas possibilidades e a lista de staff que tinha sido calculado anteriormente.

Relativamente às alterações efetuadas nas heurísticas, substituímos o algoritmo ***availability_operation/5*** por ***availability_operation_changed2/5*** como mencionado anteriormente e, aquando do agendamento das cirurgias, passamos a também a inserir as cirurgias nas agendas do *staff*, tendo sempre em atenção os intervalos de tempo que estes efetivamente participam.

CONCLUSÕES

Através do desenvolvimento e análise da complexidade do algoritmo que obtém a melhor solução, podemos concluir que o tipo de soluções, que geram todas as combinações possíveis, são extremamente ineficientes à medida que a dimensão do problema aumenta.

Deste modo, a utilização de heurísticas torna-se essencial para o desenvolvimento de soluções mais eficientes. As heurísticas permitem obter uma solução válida em um intervalo de tempo muito mais curto, mesmo que não seja necessariamente a melhor solução.