Relatório de possibilidade técnica

Criação de ofertas que minimizem o tempo dos alunos na UnB
Nível de dificuldade: 5
Dificuldades técnicas: Disponibilidade dos dados, Quantidade dos dados, criação de heurística.

Criação de grades que minimizem o tempo dos alunos na UnB
Nível de dificuldade: 3
Dificuldades técnicas: Disponibilidade dos dados.

Identificar hotspots de aglomeração utilizando visão computacional
Nível de dificuldade: 3
Dificuldades técnicas: Entender o modelo existente, necessidade de metadados ou GPS.

Liberação de turmas antes do horário previsto para evitar picos de interação social
Nível de dificuldade: 1
Dificuldades técnicas: Algoritmo de escolha tempo x professores, adesão alta de professores.

Informar a infecção de um aluno
Nível de dificuldade: 1
Dificuldades técnicas: Impossível sem acesso a relação de alunos matriculados em todas as turmas. Com esses dados, as dificuldades seriam: adesão de professores, incentivo ao abono de faltas via justificativa de contração de COVID-19 com provas (para evitar fraude).

Distribuição de marmitas do RU (sem adesão do RU)
Nível de dificuldade: 4
Dificuldades técnicas: Necessitaria de adesão muita para realmente resolver um problema, envolveria modelos estatísticos para extrapolação da lotação do RU a partir dos dados dos alunos cadastrados.

Distribuição de marmitas do RU (com adesão do RU)
Nível de dificuldade: 2
Dificuldades técnicas: Treinamento dos funcionários do RU para usar o app.

Bolhas sociais
Nível de dificuldade: 5
Dificuldades técnicas: A dificuldade é social/cultural: é muito difícil criar bolhas dentro do contexto universitário e mesmo criando essas bolhas, elas seriam muito grandes, a ponto de serem quase inúteis.

Sugestão de caminhos
Nível de dificuldade: 3.5
Dificuldades técnicas: Mapeamento da UnB em um grafo conexo com os caminhos mais usados sendo arestas e os destinos principais os nodos, imprevisibilidade do usuário em de fato seguir a recomendação e para determinação de tempo de deslocamento e determinação de heurística baseado em nível de aglomeração e agilidade do caminho, necessita de adesão alta ou de câmeras estratégicas na UnB.

Sugestão de lugares de almoço
Nível de dificuldade: 1
Dificuldades técnicas: Necessitaria de adesão muita para realmente resolver um problema. Levantamento de pontos populares de almoço. Dado adesão alta, o problema não é difícil.

//

Relatório de algoritmo/backend

Pré Requisitos: Gerador de oferta de turmas falsa. Usa gerador de números pseudo-aleatórios para criar uma oferta para que possamos testar o algoritmo, já que não teríamos tempo de extrair esses dados do matriculaweb durante a competição.

Algoritmo: Considere que o aluno diz que vai fazer as matérias “CÁLCULO1”, “FÍSICA1” e “INTRODUÇÃOÀALGEBRALINEAR”. Sejam n_c o número de turmas de “CÁLCULO1”, n_f as de “FÍSICA1” e n_ial as de “INTRODUÇÃOÀALGEBRALINEAR”. Chame n_max = max{n_c, n_f, n_ial}. Primeiro usamos a biblioteca “trotter” para pseudo-gerar todas as sequências de 3 números, com repetição e considerando a ordem, entre 0 e n_max, aberto (a biblioteca chama isso de “Amalgam”). Em seguida, descartamos aquelas combinações cuja primeira entrada seja maior que n_c, segunda entrada maior que n_f e terceira entrada maior que n_ial, uma vez que elas não correspondem à uma disponibilidade de turmas. Em seguida, associamos a cada sequência restante suas respectivas turmas e testamos se os horários colidem. Assim que uma colisão é detectada, essa turma é descartada. As sequências restantes são grades horárias válidas. Elas são avaliadas por uma heurística e o algoritmo seleciona as turmas que minimizem essa heurística.

Heurística: Inicialmente a heurística considera o número de horas que o aluno passa sem aulas na UnB e o número de dias que ele precisa sair de casa para ir para a UnB. Sejam n_idle e n_days esses números, respectivamente. Observe que n_idle é, no máximo, 55 (o aluno teria o primeiro horário de aula da UnB, que acaba às 10:00 e o último horário que começa às 21:00, totalizando 11h na UnB sem aulas. Se ele tiver essa rotina 5 vezes por semana, ele passaria 55h total na UnB sem aulas) e que o máximo de n_days é 5. A heurística então é computada como

$$heuristics = n\_days + n\_idle / 11,$$

de forma que cada parâmetro contribui, no máximo, 5 pontos para a heurística. Essa heurística pode ser modificada de acordo com a preferência do usuário de ir mais dias para UnB versus ficar menos tempo sem aulas lá.

Otimização: Apesar de performar relativamente rápido, o código ainda possui bastante espaço para melhoria. Em particular, o código foi escrito em Dart e está, para o protótipo, rodando no frontend. Rodá-lo no backend em um bom servidor e, por conseguinte, poder implementá-lo em C/C++ iria melhorar muito sua performance, em especial podendo ter um controle melhor da alocação de melhora. Outra campo onde o algoritmo pode ser otimizado é na checagem de intersecção entre conjuntos, uma vez que a API do Dart checa todas as intersecções possíveis enquanto nós só precisamos achar uma colisão para poder descartar uma grade.