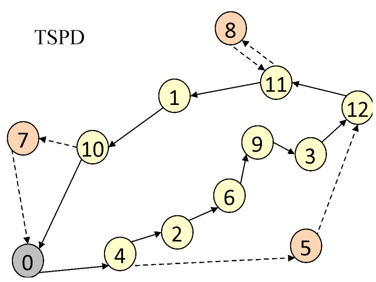
Suposições Comuns dos problemas TSPD e FSTSP

1. Ambos os veículos (caminhão e drone) iniciam a viagem no depósito e devem retornar ao depósito da ao viagem após concluir todas as entregas.
2. Em cada operação <i, j, k>, o drone só pode visitar um cliente antes de retornar ao caminhão ou depósito. Enquanto o drone está em voo, o caminhão pode fazer várias entregas.
3. O drone é lançado apenas nos locais do cliente ou depósito 0 e pousa apenas nos locais do cliente ou depósito 0'.
4. Cada cliente é visitado apenas uma vez pelo drone ou pelo caminhão. Enquanto os clientes são atendidos pelo caminhão, o drone pode estar a bordo do caminhão ou em voo.

Suposições específicas do problema:

Nós de lançamento e pouso: Em ambos os problemas TSPD e FSTSP, o drone pode ser relançado do mesmo ponto do veículo onde pousou. No entanto, no FSTSP, os drones não podem pousar no mesmo nó de onde são lançados, mas no TSPD isso é permitido.



No TSPD, o drone pode pousar para esperar pelo caminhão, mas não no FSTSP. No FSTSP o drone deve permanecer em <u>vôo constante</u> se estiver esperando o caminhão no ponto de encontro.

Tempos de preparação: o TSPD desconsidera tempo de coleta, tempo de entrega e tempo de recarga (troca de bateria). O FSTSP inclui **tempo de preparação** denotado s_L antes de um lançamento para troca de bateria e carregamento da carga e **tempo de recuperação** ao pousar no caminhão denotado por s_R.

Nós elegíveis para drones: No TSPD, todos os clientes podem ser visitados e atendidos por qualquer um dos veículos; no entanto, no FSTSP, alguns clientes não podem ser visitados por drone por vários motivos. Pode ser devido a um pacote pesado que não pode ser transportado por um drone, à necessidade de uma assinatura ou a um local de pouso impraticável.

Alcance de voo: O drone tem um tempo de voo limitado devido à capacidade limitada da bateria. Agatz et ai. (2018) resolve o problema tanto sob a premissa de **resistência de voo ilimitada** quanto **limitada**. No TSPD, a restrição de alcance de vôo contém apenas o tempo que um drone voa entre os nós.
No FSTSP considera o tempo que um drone voa entre os nós e também inclui o tempo que o drone deve permanecer em <u>vôo constante</u> se estiver esperando o caminhão no ponto de encontro, pois <u>não é permitido pousar</u> para esperar o caminhão. Pela mesma razão, cada operação FSTSP deve aderir à mesma **restrição de resistência** de voo para o caminhão. Para maior clareza, seja e a resistência do drone.
Para a operação $<i, j, k>$, a restrição TSPD relevante para o drone é apenas:

$$\tau^{d}_{i,j} + \tau^{d}_{j,k} \le e$$

Já no caso do FSTSP, a restrição para o drone é:

$$\tau^{d}_{i,j} + \tau^{d}_{j,k} + s_R \le e$$

E para o caminhão, é:

$$\tau^{tr}_{i \to k} + s_T \le e$$

onde $\tau^{tr}_{i \to k}$ é o tempo de viagem do caminhão de i a k enquanto faz entregas no meio, e s_T é o **tempo de serviço**. Se o drone é recuperado e é relançado no nó k, então $s_T = s_R + s_L$, caso contrário $s_T = s_R$ (o caminhão apenas recupera o drone sem relançá-lo no mesmo local)

Decodificação e Avaliação de Cromossomos por Programação Dinâmica

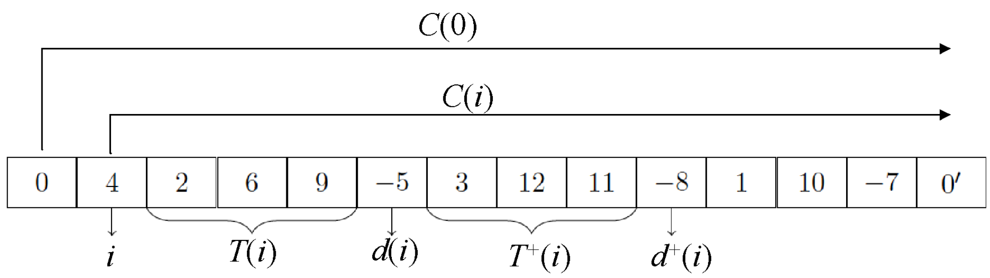
O objetivo é propor uma abordagem de programação dinâmica para determinar os **melhores pontos de lançamento** e **pouso** de acordo com a sequência e o tipo de veículos utilizados em cada nó.

Como um **sub-problema**, vamos nos referir a $C(i)$ como o menor tempo do nó i do caminhão até o final:
Ao formular uma formulação de programação dinâmica eficiente que reduz todo o problema a esses subproblemas, o objetivo ótimo pode ser alcançado.

O <u>tempo total mínimo</u> será representado por $C(0)$.

Notações:

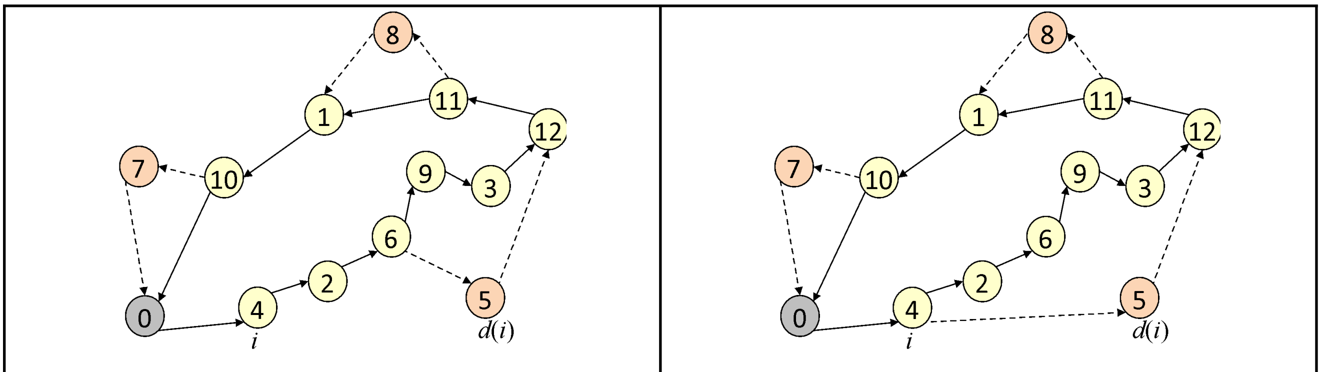
$\tau^{tr}_{i \to k}$ = Tempo de viagem do caminhão do nó i até o nó k (passando por todos os nós intermediários).
$\tau^{dr}_{i,j}$ = Tempo de viagem do drone do nó i ao nó j.
i : nó atual do caminhão.
$C(i)$: Menor tempo do sistema de caminhão e drone do nó i até o depósito 0'.
$d(i)$: Nó atendido por drone mais próximo substituindo o nó i. Se não há nenhum nó, então seria um nó fictício após o depósito.
$d^{+}(i)$: O nó atendido por drone substituindo $d(i)$. (Se nenhum, então seria um nó fictício.)
$T(i)$: O conjunto de nós atendidos pelo caminhão entre i e $d(i)$.
$T^{+}(i)$: O conjunto de nós atendidos pelo caminhão entre $d(i)$ e $d^{+}(i)$.



Em cada ponto i ($i = 0, 1, ..., n$), o objetivo é determinar o **melhor movimento** com base nas informações históricas e selecionar uma das duas opções (decisões):

- **MT (mover o caminhão)**: mover o caminhão de i para um nó de $T(i)$ enquanto o drone estiver a bordo, ou
- **LL (lançar e pousar)**: lançar o drone de i para visitar $d(i)$ e pousar em um nó de $T^{+}(i)$ enquanto o caminhão visita todos os nós intermediários.

Para o TSPD, o último nó de $T(i)$ deve ser adicionado a $T^{+}(i)$, pois o drone pode ser lançado, visitar um nó e pousar, enquanto o caminhão permanecer parado.



MT (mover o caminhão) | LL (lançar e pousar)

$\tau^{tr}_{4\to 6} = \tau^{tr}_{4,2} + \tau^{tr}_{2,6}$;

$\tau^{tr}_{4\to 12} = \tau^{tr}_{4,2} + \tau^{tr}_{2,6} + \tau^{tr}_{6,9} + \tau^{tr}_{9,3} + \tau^{tr}_{3,12}$

A transição de estados acontece da seguinte forma:

1. Seja $C_{\mathrm{MT}}(i)$ o **menor tempo** a partir do nó i onde a decisão inicial é mover o caminhão de i enquanto o drone estiver a bordo. Assim, a recursão necessária seria:

$$C_{\mathrm{MT}}(i) = \begin{cases} \infty & \text{if } \mathcal{T}(i) = \emptyset, \\ \min\limits_{k \in \mathcal{T}(i)} \{\tau^{\mathrm{tr}}_{i\to k} + C(k)\} & \text{otherwise.} \end{cases}$$

2. Da mesma forma, seja $C_{\mathrm{LL}}(i)$ o **menor tempo** possível a partir do nó i, onde a primeira decisão é enviar o drone de i para servir $d(i)$ e depois pousar no caminhão em algum nó em $T^{+}(i)$.
Com base nas diferentes suposições do problema TSPD e FSTSP, devemos ter diferentes funções para calcular $C_{\mathrm{LL}}(i)$.

- Para TSPD, seja $E^{+}(i) = \{k \in T^{+}(i) : \tau^{dr}_{i,d(i)} + \tau^{dr}_{d(i),k} <= e\}$, um subconjunto de $T^{+}(i)$ onde cada nó em $E^{+}(i)$ pode formar uma operação viável de drone com i e $d(i)$.
Portanto, a recursão necessária para TSPD será:

$$C_{\mathrm{LL}}(i) = \begin{cases} \infty & \text{if } \mathcal{E}^{+}(i) = \emptyset, \\ \min\limits_{k \in \mathcal{E}^{+}(i)} \left\{ \max\left\{\tau^{\mathrm{tr}}_{i\to k},\ \tau^{\mathrm{dr}}_{i,d(i)} + \tau^{\mathrm{dr}}_{d(i),k}\right\} + C(k) \right\} & \text{otherwise.} \end{cases}$$

- Para o **FSTSP**, precisamos saber se ocorreu um lançamento de drone no nó k. Portanto, seja σ_k:

$$\sigma_k = \{1,\ \textit{se o drone é lançado no nó } k.\ \ 0, \qquad \textit{caso contrário.}$$

Seja o conjunto:

$$\mathcal{E}^+(i) = \{k \in \mathcal{T}^+(i) : \tau^{\text{tr}}_{i \to k} + s_R + \sigma_k s_L \le e,\ \tau^{\text{dr}}_{i,d(i)} + \tau^{\text{dr}}_{d(i),k} + s_R \le e\}$$

Assim, a recursão necessária para o problema FSTSP é:

$$C_{\text{LL}}(i) = \begin{cases} \infty & \text{if } \mathcal{E}^+(i) = \emptyset, \\ \min_{k \in \mathcal{E}^+(i)} \left\{ \max\left\{\tau^{\text{tr}}_{i \to k} + s_R + \sigma_k s_L,\ \tau^{\text{dr}}_{i,d(i)} + \tau^{\text{dr}}_{d(i),k} + s_R\right\} + C(k) \right\} & \text{otherwise.} \end{cases}$$

3. Finalmente, a recursão necessária para a transição de estado é:

$$C(i) = \text{Min}\{C_{\text{MT}}(i), C_{\text{LL}}(i)\}$$

onde $C(0') = 0$.

Portanto, o algoritmo Join encontra os *pontos de encontro*, bem como o tempo total mínimo por meio de recursão para trás (*backward recursion*). Observe que $E^+(i) = \emptyset$ e $T(i) = \emptyset$ nunca ocorreriam simultaneamente. Portanto, $C(i)$ sempre terá um **valor finito** para qualquer estado i.

O número de operações necessárias para o algoritmo Join é essencial para calcular o tempo computacional desse algoritmo. Existem $O(n)$ operações necessárias para cada nó que o caminhão deve atender.
Como o número de nós de caminhão não excede n, obtemos o seguinte resultado:

Lema 1. Usando o algoritmo Join para um determinado cromossomo onde a sequência e os tipos dos veículos são conhecidos, soluções TSPD podem ser determinadas no tempo $O(n^2)$.

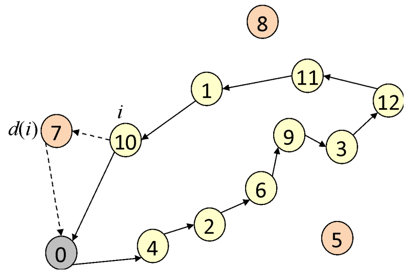
É essencial enfatizar que o algoritmo Join tem como objetivo calcular o menor tempo possível para soluções factíveis. Dado que as **soluções inviáveis** são, em última análise, desfavoráveis, devem ser penalizadas durante o processo de avaliação. O algoritmo Join será capaz de calcular o tempo mínimo para soluções inviáveis com algumas pequenas modificações. Como afirmado anteriormente, exploramos **dois tipos de inviabilidade**.
Para representações com pelo menos <u>dois nós negativos adjacentes</u>, o drone viola a premissa de uma única visita por voo. Seja w_1 a penalidade para este tipo de inviabilidade (tipo 1). O tempo de viagem para o lançamento do drone em i, visitando j_1, j_2, ..., j_m e pousando em k, será calculado como:

$$\tau^{\text{dr}}_{i,j_1} + \sum_{q=1}^{m-1} w_1^q \tau^{\text{dr}}_{j_q,j_{q+1}} + \tau^{\text{dr}}_{j_m,k}$$

Por outro lado, **se o alcance de voo for violado pelo drone** em TSPD ou por qualquer um dos veículos em FSTSP a solução é tipo 2 inviável. Com apenas algumas modificações, as mesmas recursões podem ser usadas para calcular o custo. Para começar, o conjunto $E^+(i)$ deve ser substituído pelo conjunto $T^+(i)$, uma vez que todos os movimentos são possíveis, independentemente de a restrição de alcance do drone ser ou não violada. Seja w_2 a penalidade para a inviabilidade do tipo 2.
Para TSPD, só precisamos adicionar $w_2.\max\{0, \tau^{dr}_{i,d(i)} + \tau^{dr}_{d(i),k} - e\}$ a $\tau^{dr}_{i,d(i)} + \tau^{dr}_{d(i),k}$

Para FSTSP precisamos adicionar $w_2.\max\{0, \tau^{tr}_{i\to k} + s_R + \sigma_k s_L - e\}$ a $\tau^{tr}_{i\to k} + s_R + \sigma_k s_L$ e $w_2.\max\{0, \tau^{dr}_{i,d(i)} + \tau^{dr}_{d(i),k} + s_R - e\}$ a $\tau^{dr}_{i,d(i)} + \tau^{dr}_{d(i),k} + s_R$

											$T(i) = \phi$		
0	4	2	6	9	−5	3	12	11	−8	1	10	−7	0′
											i	$d(i)$	$T^+(i)$



$C(0') = 0.$

$C(10) = \text{Min}\{C_{\text{MT}}(10), C_{\text{LL}}(10)\}$

$C_{\text{MT}}(10) = \infty$, pois $T(10) = \emptyset$

$E^+(10) = \{0'\}$

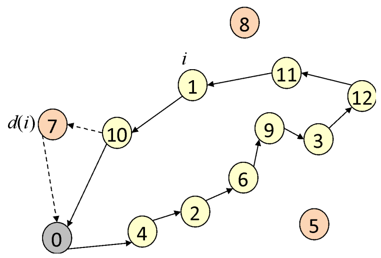
$C_{\text{LL}}(10) = Min_{k \in E^+(10)} \{ \text{Max}\{ \tau^{tr}_{10\to k} + s_R + \sigma_k s_L, \tau^{dr}_{10,7} + \tau^{dr}_{7,k} + s_R\} + C(\text{k}) \}$

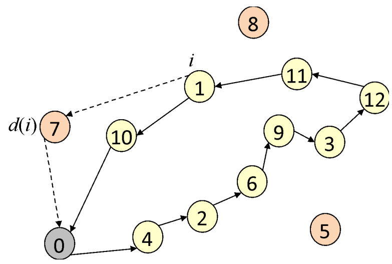
$\sigma_{0'} = 0$, pois em 0' nao será lançado o drone.

$C_{\text{LL}}(10) = \text{Max}\{ \tau^{tr}_{10\to 0'} + s_R, \tau^{dr}_{10,7} + \tau^{dr}_{7,0'} + s_R\} + C(0').$

=> $C(10) = C_{\text{LL}}(10)$

0	4	2	6	9	−5	3	12	11	−8	1	10	−7	0′
										i	$T(i)$	$d(i)$	$T^+(i)$





$C(1) = \mathrm{Min}\{C_{\mathrm{MT}}(1), C_{\mathrm{LL}}(1)\}$

$T(1) = \{10\}$

$C_{\mathrm{MT}}(1) = Min_{k\in T(1)}\{\tau^{tr}_{1\to k} + C(k)\} = \tau^{tr}_{1\to 10} + C(10) = \tau^{tr}_{1,10} + C(10)$

$E^{+}(1) = \{0'\}$

$C_{\mathrm{LL}}(1) = Min_{k\in E^{+}(1)}\{\ \mathrm{Max}\{\tau^{tr}_{1\to k} + s_R + \sigma_k s_L,\ \tau^{dr}_{i,d(i)} + \tau^{dr}_{d(i),k} + s_R\} + C(\mathrm{k})\ \}$

$\sigma_{0'} = 0$, pois em 0’ nao será lançado o drone.

$C_{\mathrm{LL}}(1) = \mathrm{Max}\{\tau^{tr}_{1\to 0'} + s_R,\ \tau^{dr}_{1,7} + \tau^{dr}_{7,0'} + s_R\} + C(0')$

$C_{\mathrm{LL}}(1) = \mathrm{Max}\{\tau^{tr}_{1,10} + \tau^{tr}_{10,0'} + s_R,\ \tau^{dr}_{1,7} + \tau^{dr}_{7,0'} + s_R\} + C(0')$.

$T(i) = \phi$

0	4	2	6	9	−5	3	12	11	−8	1	10	−7	0′
								i	$d(i)$	$\underbrace{\quad}_{T^{+}(i)}$			

$C(11) = \mathrm{Min}\{C_{\mathrm{MT}}(11), C_{\mathrm{LL}}(11)\}$

$C_{\mathrm{MT}}(11) = \infty$, pois $T(10) = \emptyset$

$C_{\mathrm{LL}}(11) = Min_{k\in E^{+}(11)}\{\ \mathrm{Max}\{\tau^{tr}_{11\to k} + s_R + \sigma_k s_L,\ \tau^{dr}_{11,8} + \tau^{dr}_{8,k} + s_R\} + C(\mathrm{k})\ \}$

Se $E^{+}(11) = \{1, 10\}$

$C_{\mathrm{LL}}(11) = Min\{\ \mathrm{Max}\{\tau^{tr}_{11\to 1} + s_R + \sigma_1 s_L,\ \tau^{dr}_{11,8} + \tau^{dr}_{8,1} + s_R\} + C(1),$

$\mathrm{Max}\{\tau^{tr}_{11\to 10} + s_R + \sigma_{10} s_L,\ \tau^{dr}_{11,8} + \tau^{dr}_{8,10} + s_R\} + C(10)\ \}$

0	4	2	6	9	−5	3	12	11	−8	1	10	−7	0′
						i	$\underbrace{\quad}_{T(i)}$		$d(i)$	$\underbrace{\quad}_{T^{+}(i)}$			