時間枠つき配送計画問題に対するメタ戦略アルゴリズム

柳浦睦憲 （京都大学）

with

橋本英樹（京都大学） 茨木俊秀（関西学院大学）

今堀慎治（東京大学） 久保幹雄（東京海洋大学）

増田友泰（アクセンチュア） 野々部宏司（法政大学）

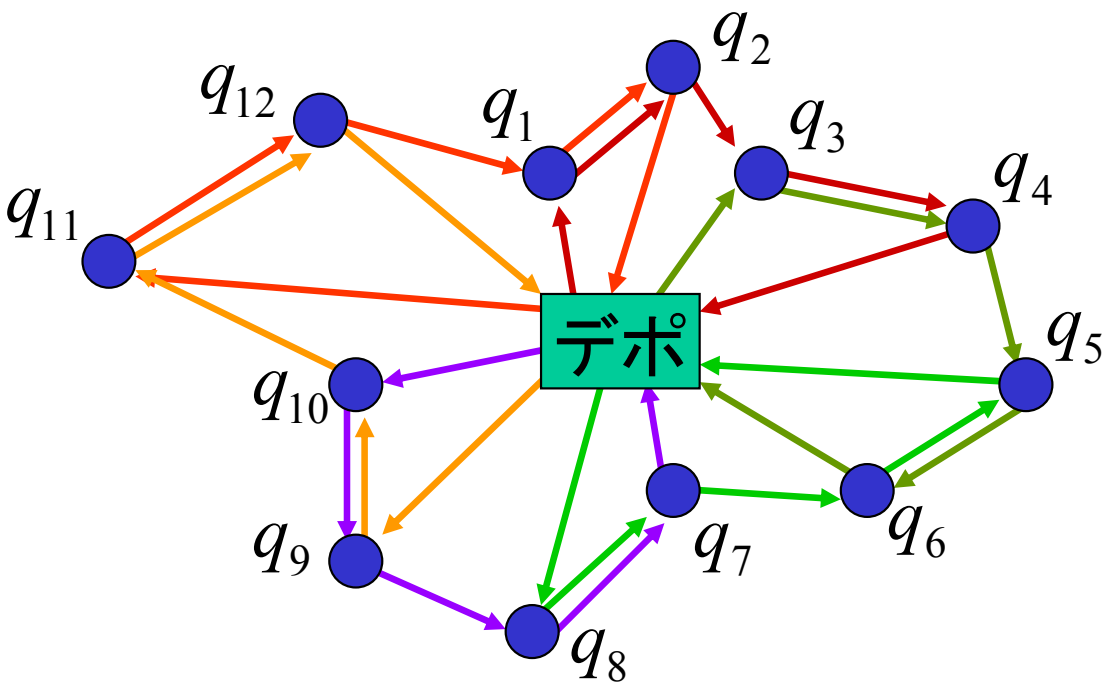
祖父江謙介（トヨタ） 宇野毅明（国立情報学研究所）

時間枠付配送計画問題

入力: 節点$V = \{0, 1, \ldots, n\}$ (0: デポ, $i \geq 1$: 客), 車両$M = \{1, \ldots, m\}$,
距離 d_{ij} , 移動時間 t_{ij} , 容量 Q_k , 重み q_i

出力: 総距離を最小にする車両ルート

制約: 容量制約 および 時間枠制約



容量制約

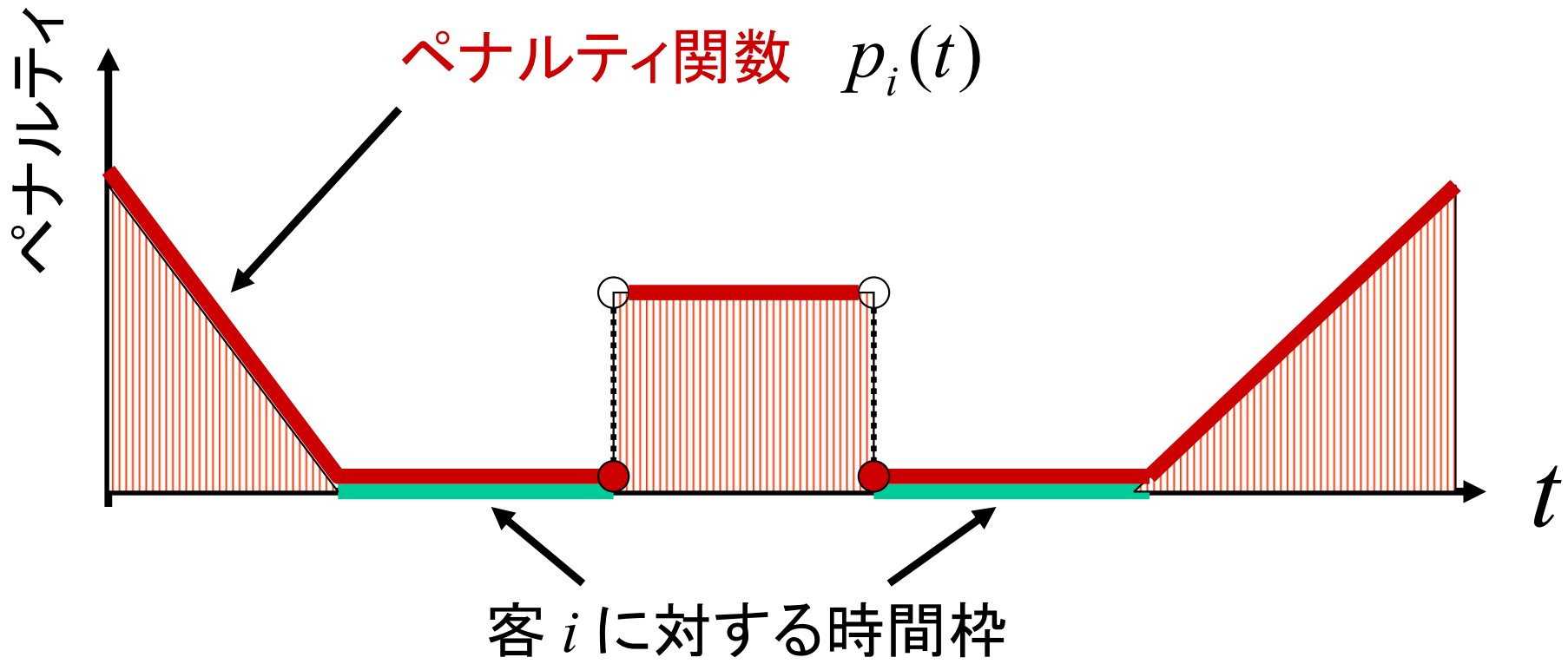
$q_1 + q_2 + q_3 + q_4 \leq Q_1$

$q_5 + q_6 + q_7 + q_8 \leq Q_2$

$q_9 + q_{10} + q_{11} + q_{12} \leq Q_3$

一般化時間枠

各客はサービスを受ける時刻に対する希望を提示する



$p_i(t)$: 非凸かつ不連続でもよい
（ただし区分線形関数のみ）

目的関数

$$cost(\sigma) = D(\sigma) + T(\sigma) + Q(\sigma)$$

σ	車両スケジュール
$D(\sigma)$	総距離
$T(\sigma)$	総時間ペナルティ
$Q(\sigma)$	総容量超過

時間枠制約および容量制約

⬇

ソフト制約

計算例

配送計画問題の基本モデルの問題例

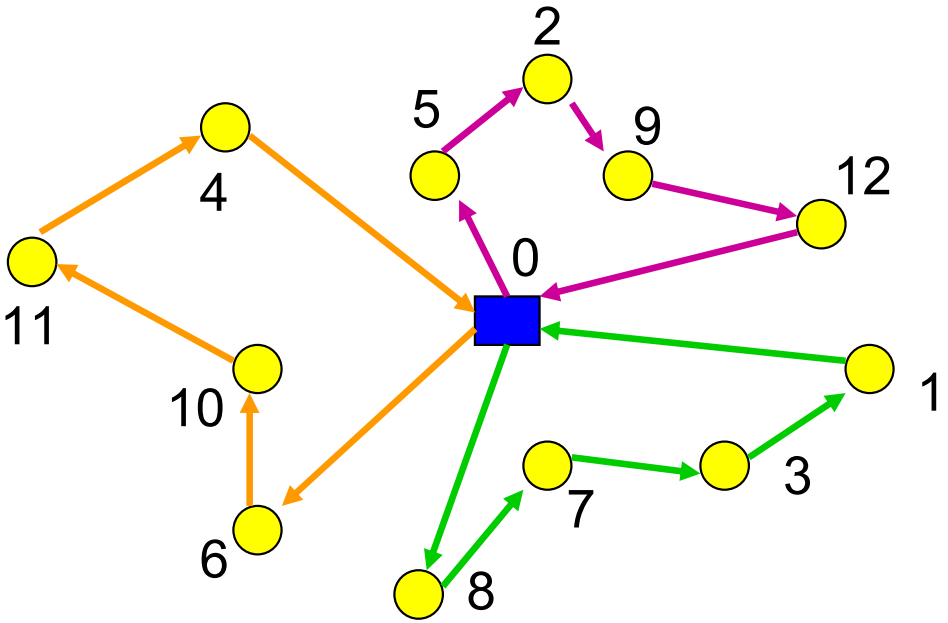
計算例１

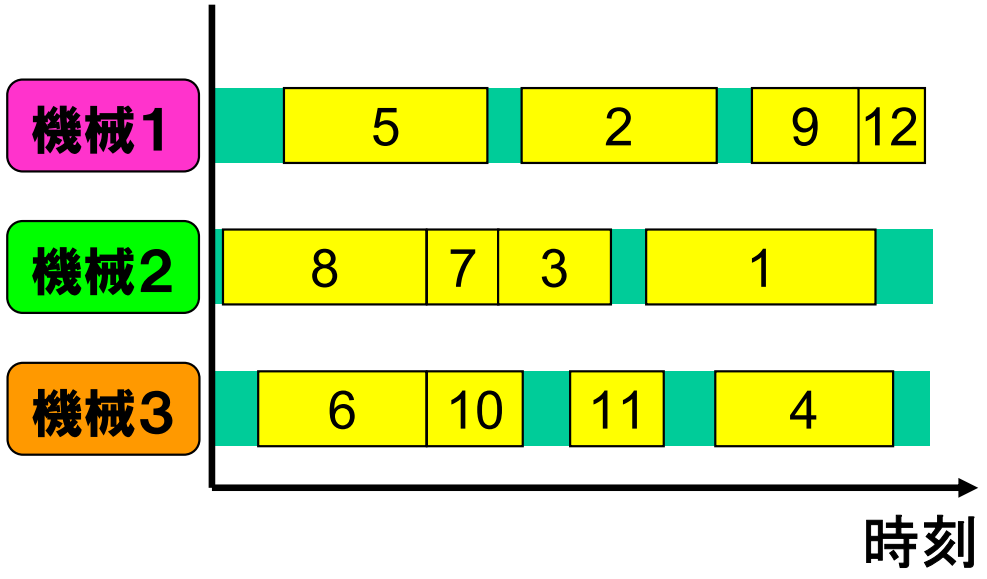
時間枠つきの問題例

計算例２

スケジューリング

客	⇔	仕事
車両	⇔	機械
移動時間	⇔	処理時間とセットアップ時間
時間枠制約	⇔	準備時間や納期など
容量制約	⇔	資源制約





問題構造と探索空間

車両スケジュール:

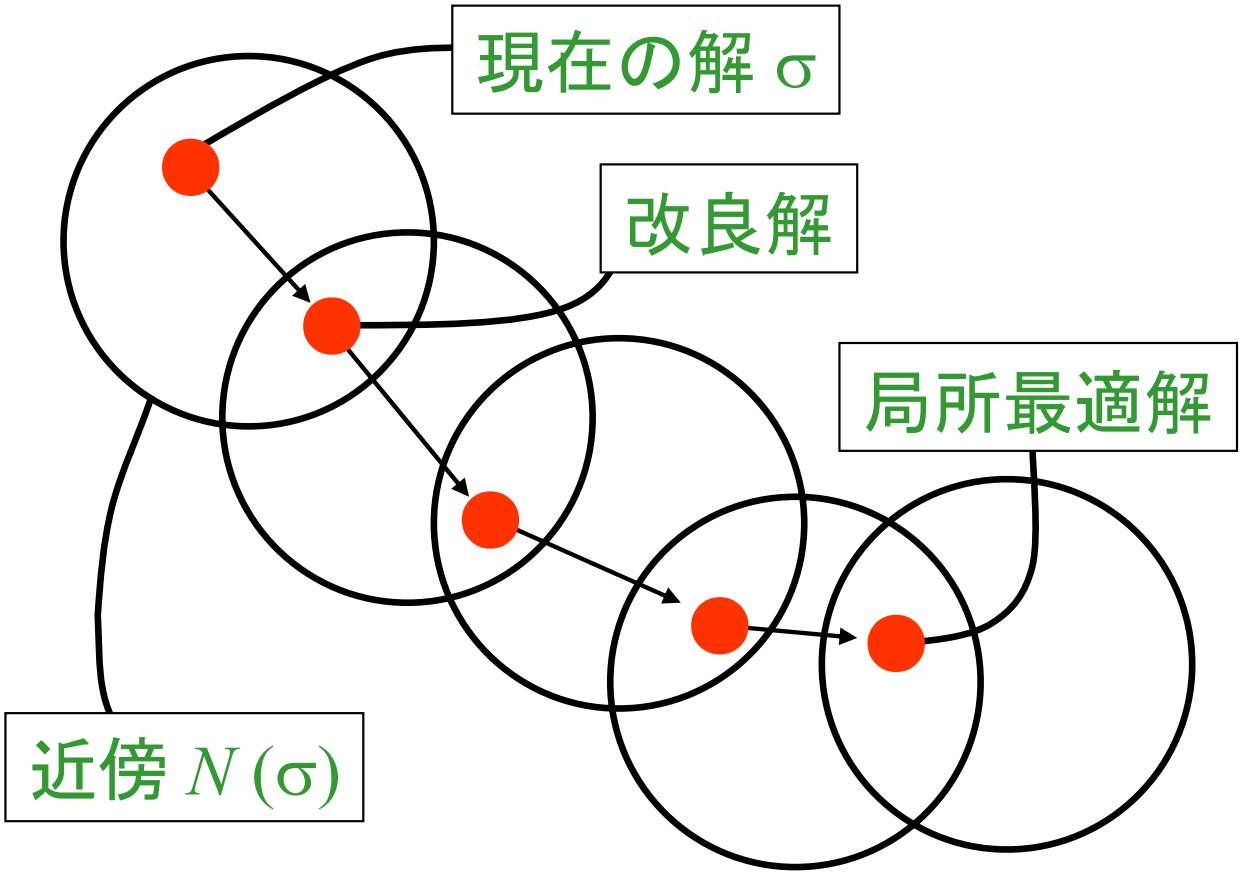
(a) 客の車両への割当
(b) 各車両における客の訪問順序
(c) 各客のサービス開始時刻

(a) と (b) → 局所探索法により探索

(c) → 動的計画法により最適に解く
((a)と(b)は固定という条件の下で)

局所探索法

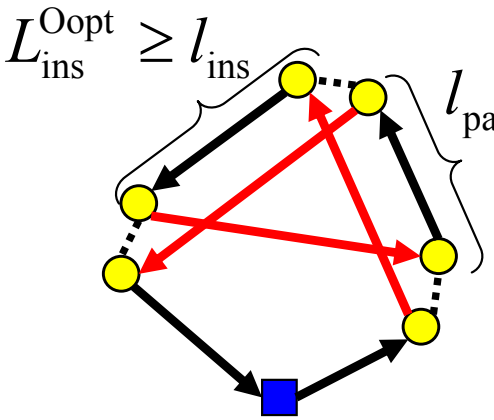
適当な初期解から始め、現在の解 σ の近傍 $N(\sigma)$ の中に σ よりもよい解 σ' があれば $\sigma := \sigma'$ と置き換える操作を反復



- 探索空間
- 解の評価法
- 近傍
- 初期解生成法

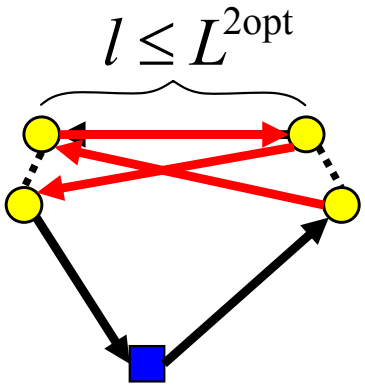
近傍操作

Or-opt



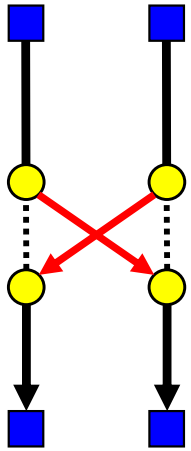
$O(L_{\text{path}}^{\text{Oopt}} L_{\text{ins}}^{\text{Oopt}} n)$

2-opt



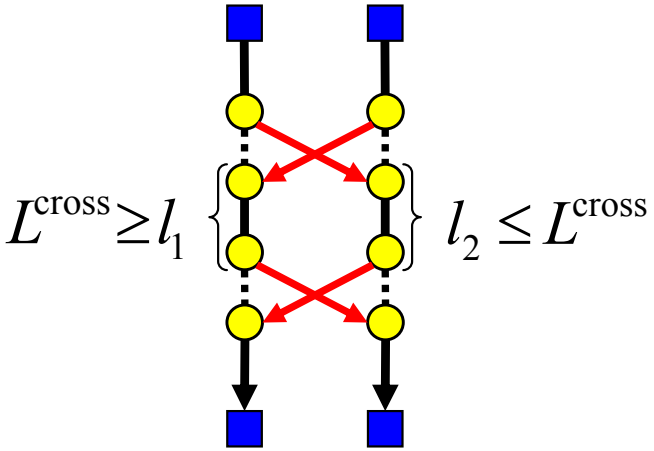
$O(L^{2\text{opt}} n)$

2-opt*



$O(n^2)$

Cross exchange



$O(\,(L^{\text{cross}})^2 n^2)$

近傍内の解: 現在の解のパスを高々4つつなぎ合わせることで生成できる

最適サービス開始時刻決定問題

もともとの目的関数　$cost(\sigma) = D(\sigma) + T(\sigma) + Q(\sigma)$

問題（車両ルートは固定）

Optimal Start Time Problem (OSTP)

入力: 車両 k の客の訪問順序

$\sigma_k(0),\ \sigma_k(1),\ \ldots,\ \sigma_k(n_k),\ \sigma_k(n_k+1)$

出力: 車両 k の時間ペナルティを最小にするような、各客のサービス開始時刻

➡ 動的計画法により厳密に最適化

OSTPの関連研究

時間ペナルティ関数が一般の場合

Ibaraki, Imahori, Kubo, Masuda, Uno & Yagiura, 2005

時間ペナルティ関数が凸の場合

- 関連研究:

  Garey, Tarjan & Wilfong 1988
  Powell & Solomon 1992
  Davis & Kanet 1993
  Taillard, Badeau, Gendreau, Guertin & Potvin 1997
  Tamaki, Komori & Abe 1999
  Ahuja & Orlin 2001
  Hochbaum & Queyranne 2003

- OSTP は線形計画問題になる
- isotonic median regression 問題を含む
- $O(n_k \log n_k)$ 時間のアルゴリズムあり

動的計画法

$f_h^k(t)$: 客 $\sigma_k(0), \sigma_k(1), \ldots, \sigma_k(h)$ が時刻 t 以前にサービスされるときのこれらの客の時間ペナルティの最小値

$$f_0^k(t) = \begin{cases} +\infty, & t \in (-\infty, e_0) \\ 0, & t \in [e_0, +\infty) \end{cases}$$

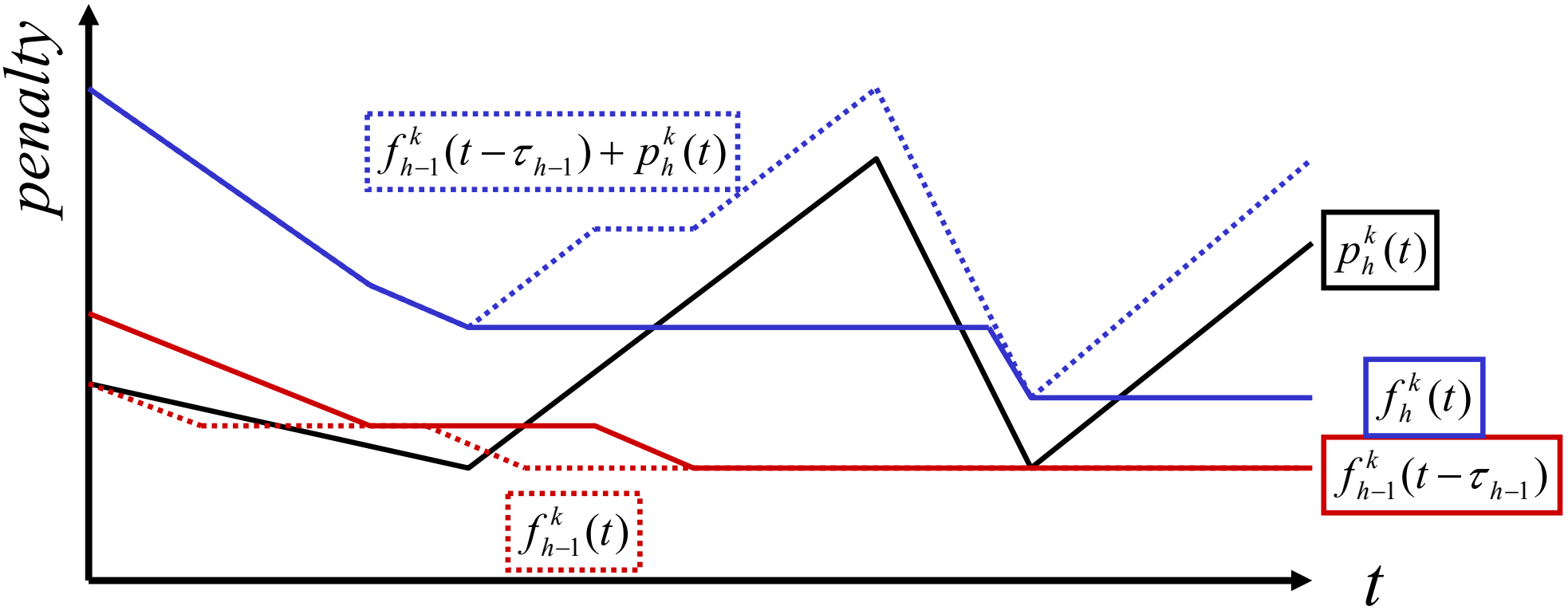
$$f_h^k(t) = \min_{t' \le t} (f_{h-1}^k(t' - \tau_{h-1}) + p_h^k(t')), \quad 1 \le h \le n_k + 1$$

e_0: 車両のデポからの出発可能時刻

τ_h: h 番目の客から $h+1$ 番目の客への移動時間

$p_h^k(t)$: h 番目の客の時間ペナルティ関数

動的計画法の実行例



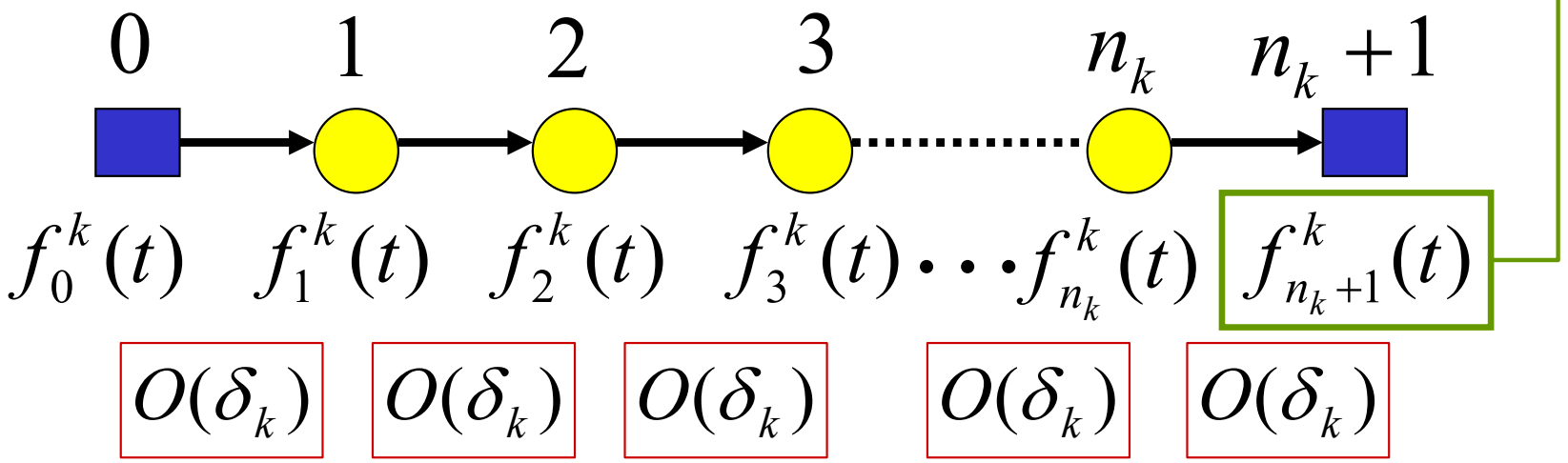
$$f_h^k(t)=\min_{t'\le t}(f_{h-1}^k(t'-\tau_{h-1})+p_h^k(t'))$$

動的計画法の計算時間

n_k: 車両 k のルート内の客数

δ_k: 車両 k のルート内の客の時間ペナルティ関数の線形区分数の合計 (通常 $\delta_k = O(n_k)$)

最適ペナルティ値決定



総計算時間: $O(n_k\delta_k)$

時間ペナルティの評価の高速化

アイデア

- 近傍内の解の評価に現在の解 σ の $f_h^k(t)$ （計算済）を再利用
- 解が更新されたときには動的計画法(DP)を再計算

ペナルティ関数が一般の場合:

近傍内の解の評価: $O(\delta_k)$

解の移動の際のDPの再計算: $O(n_k \delta_k)$

ペナルティ関数が凸の場合:

近傍内の解の評価: $O(\log \delta_k)$

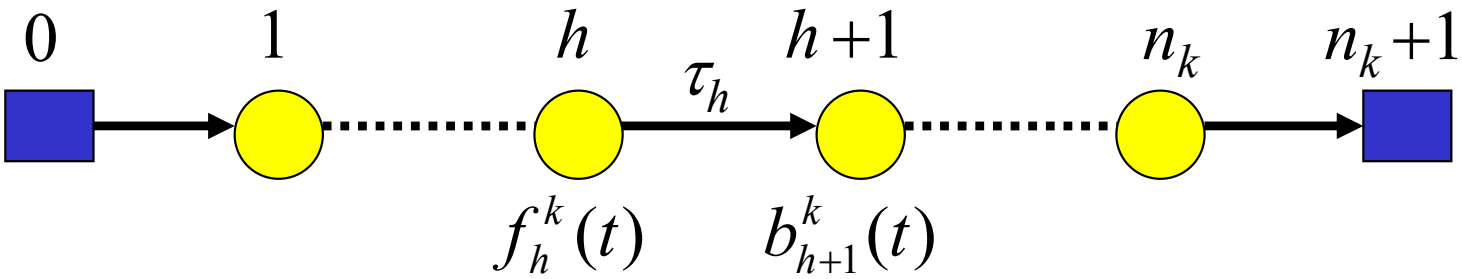
解の移動の際のDPの再計算: $O(\delta_k \log \delta_k)$

平衡木を利用

時間ペナルティの高速計算

$b_h^k(t)$: 客 $\sigma_k(h), \sigma_k(h+1), \ldots, \sigma_k(n_k+1)$ が時刻 t 以降にサービスされるときのこれらの客の時間ペナルティの最小値

$f_h^k(t)$ と同様に計算できる



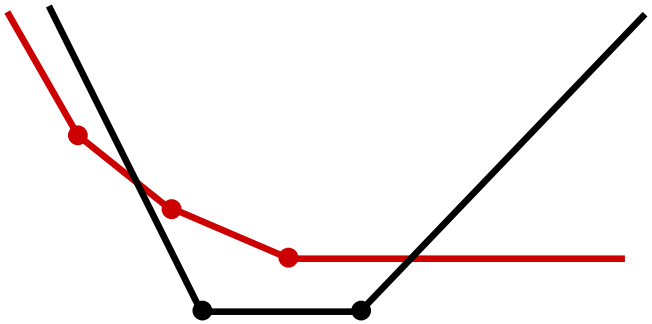
ルート k の最小ペナルティ: $\min_t \left(f_h^k(t) + b_{h+1}^k(t+\tau_h) \right)$

- ペナルティが一般の場合$O(\delta_k)$, 凸の場合$O(\log \delta_k)$時間
- 2-opt*には直接利用可. それ以外の場合は交換されるパス内の客の$f_h^k(t)$を効率よく再計算できるよう探索順序に工夫の必要あり.

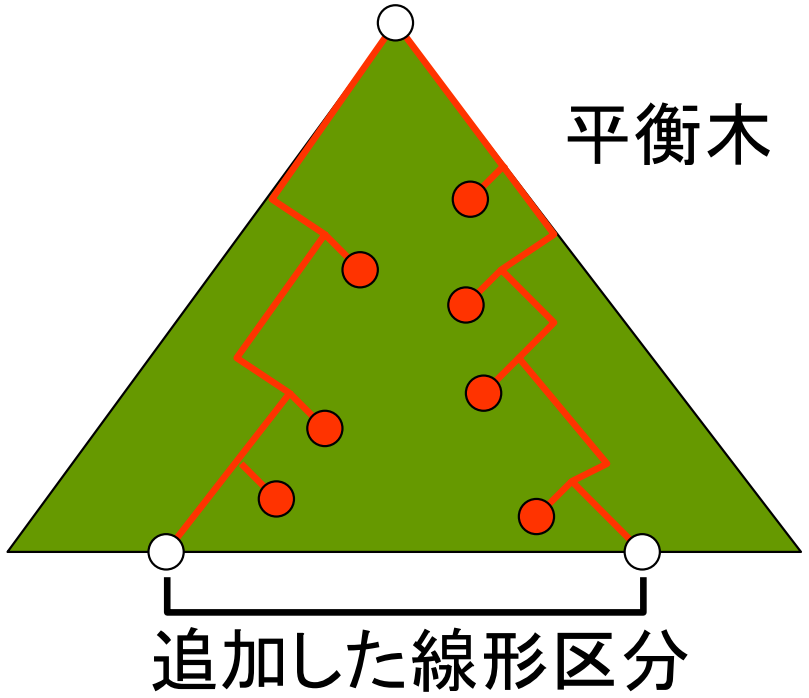
DP: 時間ペナルティが凸の場合

$p_i(t)$: 全ての i に対して凸 $\Rightarrow f_h^k(t)$: 凸

$f_h^k(t)$ を平衡木で表現
- 葉節点 $\leftrightarrow f_h^k(t)$の線形区分のひとつ
- 根から葉へのパス
  $\leftrightarrow$ 葉に対応する線形区分の傾きや切片などの情報



$f_{h-1}^k(t-\tau_{h-1})+p_h^k(t)$



基本操作: $O(\log \delta_k)$ 時間
総計算時間: $O(\delta_k \log \delta_k)$

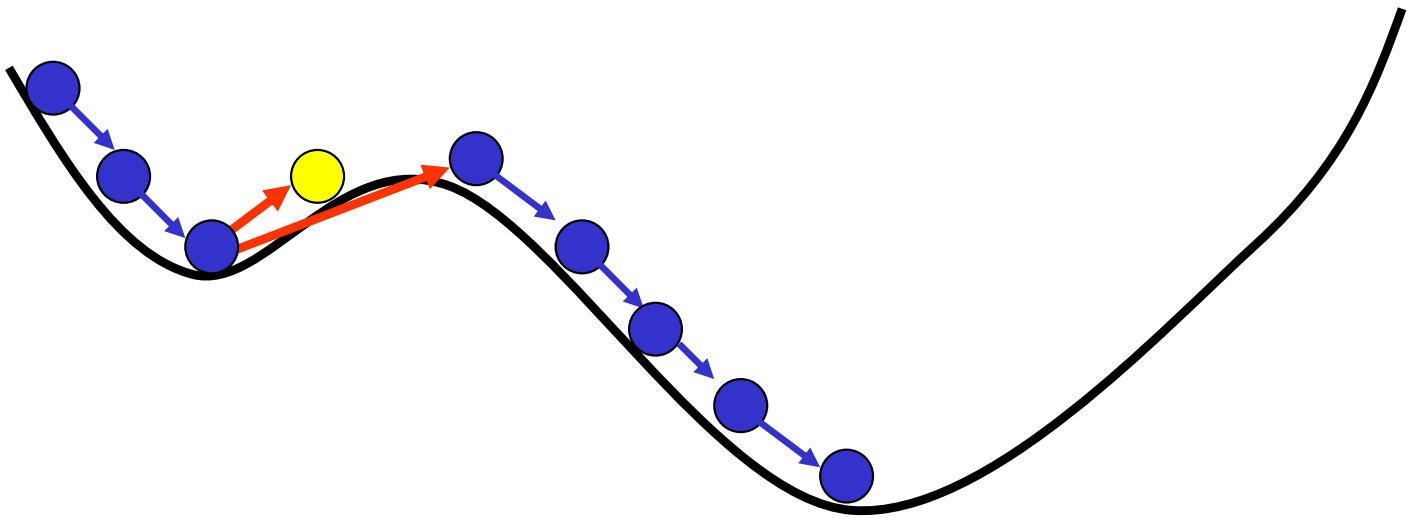
効率化のアイデア

- 近傍の制限
  - 近傍リスト: TSP や VRP に対し大きな効果
  - サービス開始時刻が近い客のペア
- 動的計画法の関数 $f_h^k(t)$ の区分数の削減
  - 関数 $f_h^k(t)$ の線形区分の中で対応する

    ペナルティ値が非常に大きいものを削除

反復局所探索法 (ILS)

- 局所探索を多数反復
- 過去の探索で得られたよい解に少しのランダムな変形を加えたものを初期解



計算実験

問題例

- Gehring & Homberger のベンチマーク問題例
  （Solomonの問題例と同様の生成法．より大規模）
- 時間枠制約と容量制約: 絶対制約
- 客数: 200, 400, 600, 800, 1000
- 各サイズに対して60問（全部で300問）

計算環境と実験方法

- C 言語; PC（Pentium IV 2.8GHz）
- 絶対制約 → ペナルティ関数; 適応的重み調整
- 出力する解: 絶対制約を満たす解
- 車両数: 既知の最良値（± 1）

number of customers		best known	MB (2003)	GH (2002)	LC (2003)	our ILS
200	CNV	692	694	696	694	694
	CTD	169281	168537	179328	173061	170331
	time (min)		5.88	3.83	21.66	33.3
400	CNV	1386	1389	1392	1390	1384
	CTD	392444	390386	428489	408281	401285
	time (min)		12.49	12.95	43.32	66.6
600	CNV	2076	2082	2079	2088	2070
	CTD	799355	796172	890121	836261	827192
	time (min)		29.39	23.53	64.97	100
800	CNV	2754	2760	2765	2766	2750
	CTD	1429914	1535849	1361586	1475281	1426133
	time (min)		42.32	106.53	86.63	133.3
1000	CNV	3438	3446	3446	3451	3434
	CTD	2106125	2078110	2290367	2225366	2169452
	time (min)		440.82	361.2	108.29	166.6

CNV: cumulative number of vehicles

CTD: cumulative total distance

MB: Mester and Bräysy

GH: Gehring and Homberger

LC: Le Bouthillier and Crainic

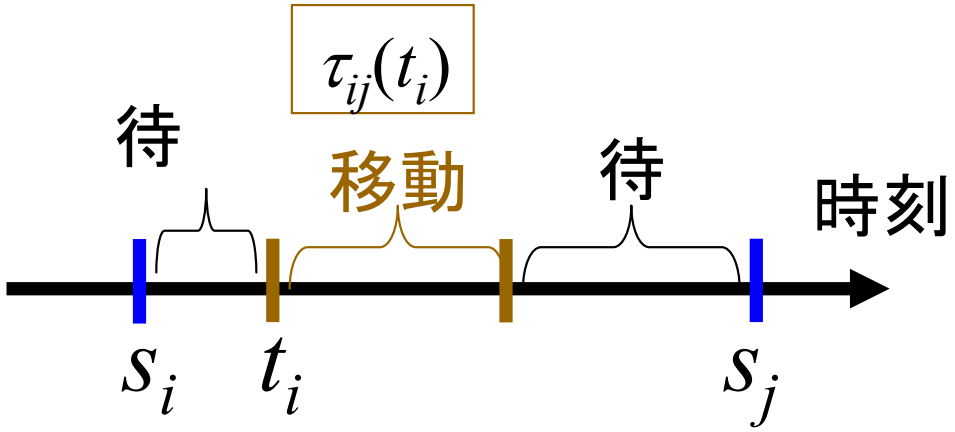
より高い汎用性を目指して

- 時刻依存の移動時間および移動コスト
  - 朝夕のラッシュ
  - 生産ラインにおけるセットアップ

- 移動時間が可変，移動時間に応じたコスト
  - 有料道路の利用により移動時間を短縮
  - 特急仕事として高速処理

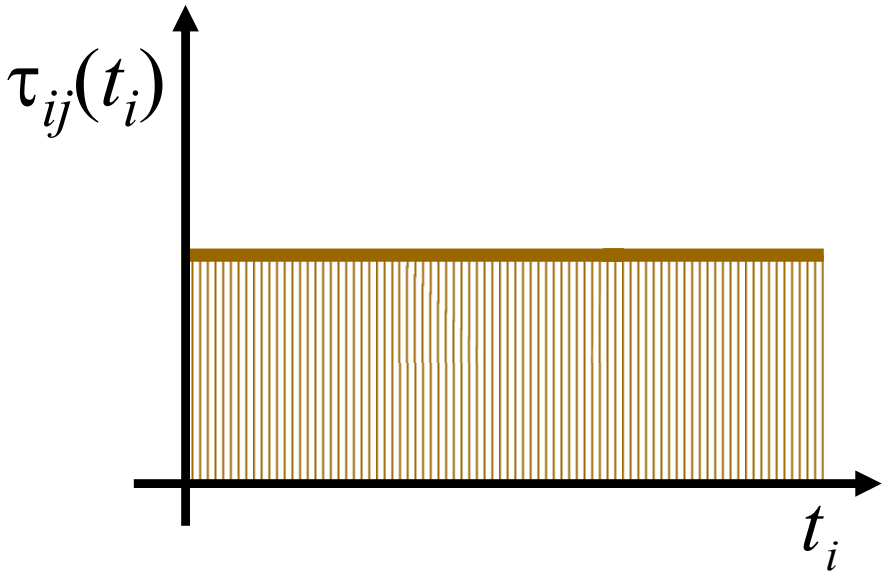
→ 一般の時間ペナルティをもつ問題に上の
いずれかを追加した問題を考える

時刻依存の移動時間 $\tau_{ij}(t_i)$

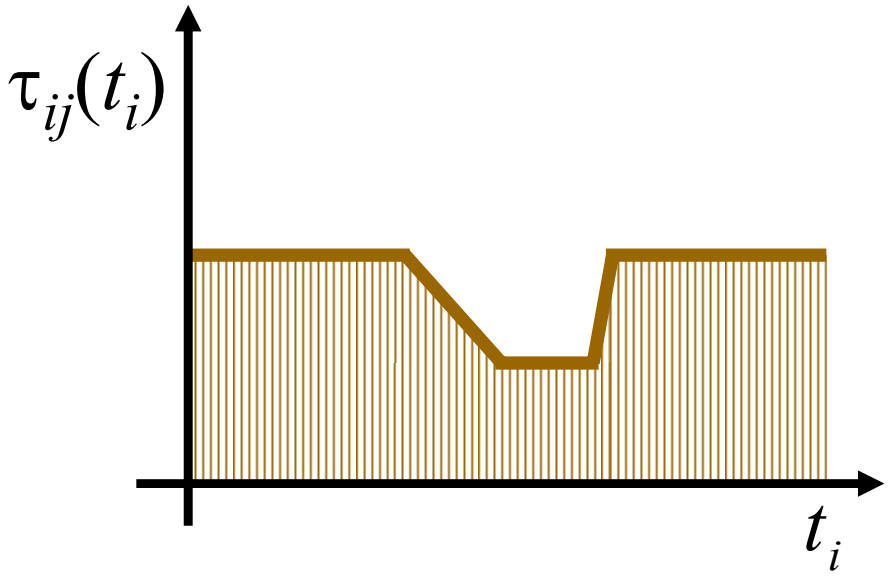


- 連続
- FIFO

$$t < t' \Rightarrow t + \tau_{ij}(t) < t' + \tau_{ij}(t')$$

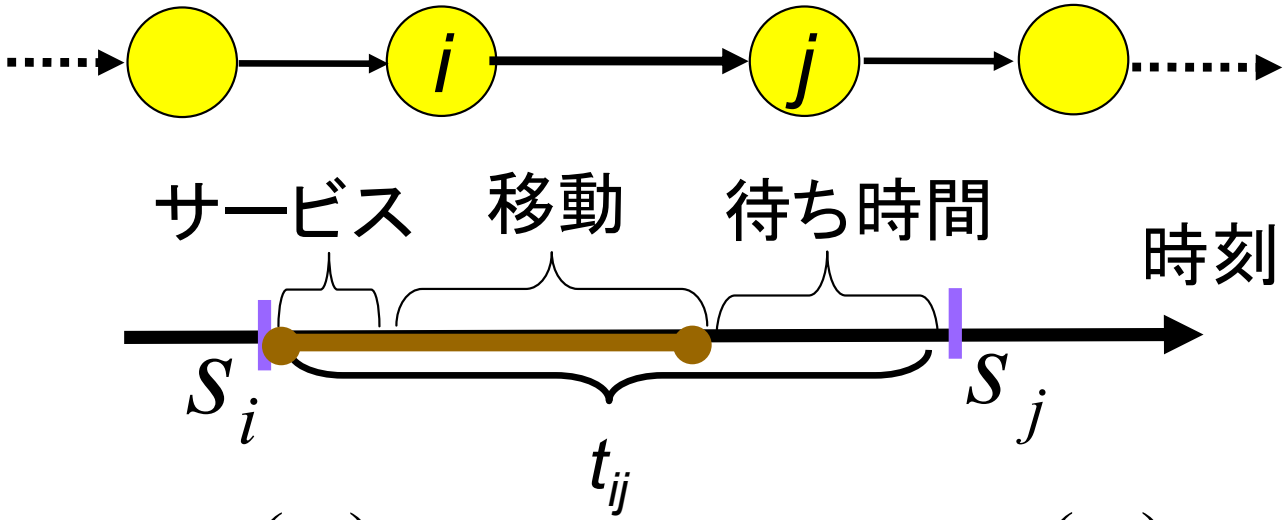


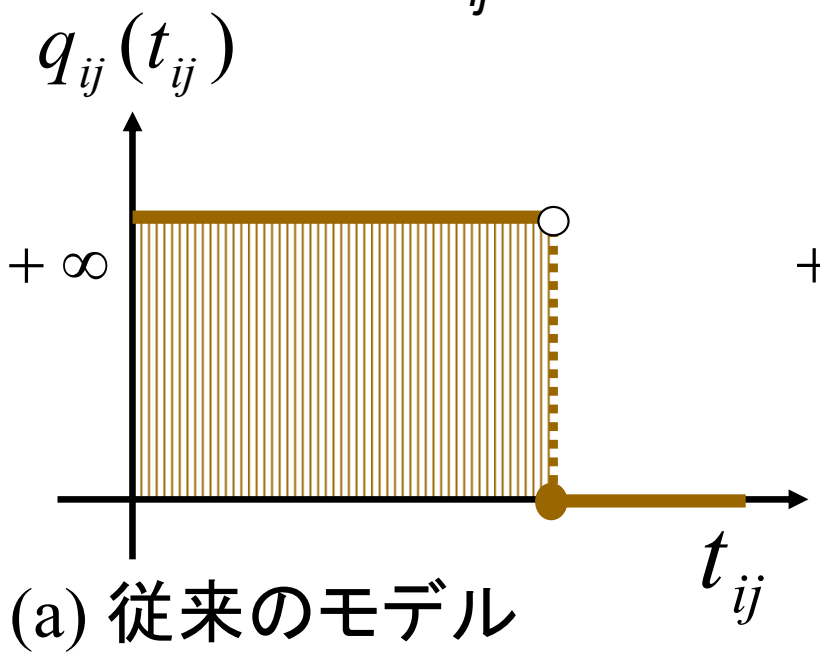
(a) 従来のモデル

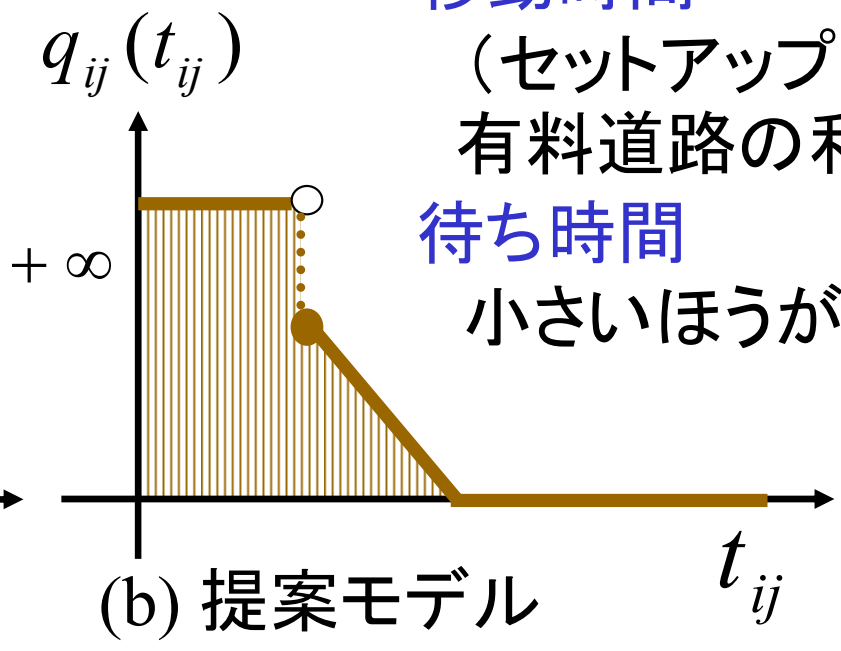


(b) 提案モデル

移動時間コスト関数 $q_{ij}(t_{ij})$







サービス時間
（処理時間）
人手やコストをかけて
短縮可能

移動時間
（セットアップ時間）
有料道路の利用など

待ち時間
小さいほうがよい場合

計算の複雑さに関する結果

- 時刻依存の移動時間および移動コスト
  - 移動時間関数の総複雑度に対して線形時間のDP
  - 移動時間関数の複雑度が各客ペアでO(1)
    → 移動時間が定数の場合と同程度の計算時間

- 移動時間が可変，移動時間に応じたコスト
  - 移動時間コスト関数が一般
    → NP困難; 擬多項式時間のDP
  - 移動時間コスト関数が凸（時間ペナルティ関数は一般）→ 多項式時間のDP

まとめ

- 時間枠に対する要求を一般的なペナルティ関数として表現できる汎用的な問題の定式化
  - → 時間ペナルティ最小化問題を解く必要性
  - → 動的計画法の提案
- 高速化のさまざまな工夫
- Gehring & Homberger のベンチマーク問題例に対する良好な結果
- 移動時間に対する汎用的な定式化

今後の課題

- 異なるタイプの汎用化
- 配送ルートの探索能力の向上
- 大規模問題例への適用
- 時間枠の取り扱い　→

  より実装しやすい単純なアルゴリズムで

  高速かつ実用的なものの開発

  cf. 提案手法はいずれも非常に複雑で実装困難