理論輪読会

J. ホロムコヴィッチ: 計算困難問題に対するアルゴリズム理論

3章前半 (3.2 – 3.4)

20140622
社会基盤学専攻 交通研 伊藤篤志

3.1 序論

NP問題を解く際に考える3つのアプローチ

①困難問題の集合を, 易しい入力インスタンスと困難な入力インスタンスに分割

②増加率が非常に遅い最悪指数時間計算量のアルゴリズムを設計

ex) $Time_A(n)=2^{\sqrt{n}}$, $(1.2)^n$

③問題の条件緩和

ex) 上下界を求める

擬多項式時間アルゴリズム

アプローチ①に基づく考え方

困難な入力インスタンス: 入力サイズの指数時間
易しい入力インスタンス: 多項式時間
→入力整数値の値が入力整数の個数に対して多項式

整数値問題

入力が整数の集合と見なされる問題

整数値問題に対する擬多項式時間アルゴリズム:

実行時間が入力整数の個数と入力整数値の多項式

$|I|$　　$Max\text{-}Int(x)$

ナップサック問題 (KP)

目的関数: 総コスト
制約条件: 荷物の総荷重

$$\max \sum_{i \in T} c_i$$

$$\sum_{i \in T} w_i \leq b$$

$T \subseteq \{1,2,\ldots,n\}$
(w_i, c_i) = (重さ, コスト)

KPの問題インスタンスI: $I=(\underbrace{w_1, w_2,\ldots, w_n}_{n個}, \underbrace{c_1, c_2,\ldots, c_n}_{n個}, b)$

動的計画法により, Iの解を効率よく計算するため, I_iの部分集合のみ計算

部分インスタンス$I_i=(w_1,\ldots, w_i, c_1,\ldots, c_i, b)$

各I_i $(i=1,2,\ldots,n)$と各整数$k \in \{0,1,\ldots,\Sigma^n_{i=1}c_i\}$に対して, 以下の3項組を計算

$$(k, W_{i,k}, T_{i,k}) \in \left\{0,1,2,\ldots,\sum_{j=1}^{i} c_j\right\} \times \{0,1,2,\ldots,b\} \times Pot(\{1,2,\ldots,i\})$$

$$k = \sum_{j \in T_{i,k}} c_j, W_{i,k} = \sum_{j \in T_{i,k}} w_j, T_{i,k} \subseteq \{1,2,\ldots i\}$$

アルゴリズム3.2.2.2(DPKP)

入力値 $I=(w_1, w_2,..., w_n, c_1, c_2,..., c_n, b)$, nは正整数

KP: $I=(w_1,..., w_5, c_1,..., c_5, b)$

i	1	2	3	4	5
w	23	15	15	33	32
c	33	23	11	35	11
b			65		

Step 1 $TRIPLE_1 = \{(0, 0, \Phi) \cup (c_1, w_1, \{1\}) | w_1 \leqq b\}$ $O(1)$

$TRIPLE_1 =$
$\{(0, 0, \Phi) \cup (33, 23, \{1\})\}$

Step 2 $O(n\text{-}1)$

for i (i=1 to n-1)
  $SET(i+1)=TRIPLE_i$
  for 各$(k, w, T) \in TRIPLE_i$ $O(|TRIPLE_i|)$
    $SET(i+1)=$
    $SET(i+1) \cup \{(k+c_{i+1}, w+w_{i+1}, T \cup \{i+1\})\}$
$TRIPLE_{i+1} = SET(i+1)$のうち,
実現可能かつ最小重みをもつ
3項組(m, w', T')を選択

$i=i+1$

Rap 1 (i=1)

$SET(2)=TRIPLE_1$
  $SET(2)=$
  $SET(2) \cup \{(23, 15, \{2\}),$
  $(56, 38, \{1,2\})\}$

$TRIPLE_2=SET(2)$

アルゴリズム3.2.2.2(DPKP)

Step 2

for i (i=1 to n-1) $O(n\text{-}1)$

$SET(i+1)=TRIPLE_i$

for 各$(k, w, T)\in TRIPLE_i$ $O(|TRIPLE_i|)$

$SET(i+1)=$

$SET(i+1)\cup\{(k+c_{i+1}, w+w_{i+1}, T\cup\{i+1\})\}$

$i=i+1$

$TRIPLE_{i+1}=SET(i+1)$のうち,
実現可能かつ最小重みをもつ
3項組(m, w', T')を選択

Rap 2 (i=2)

$SET(3)=TRIPLE_2$

$SET(3)=$

$SET(3)\cup\{(11, 15, \{3\}),$
$(34, 30, \{2,3\})$
$(44, 38, \{1,3\})$
$(67, 53, \{1,2,3\})\}$

$TRIPLE_3=SET(3)$

アルゴリズム3.2.2.2(DPKP)

Step 2

for i (i=1 to n-1) $O(n-1)$

$SET(i+1)=TRIPLE_i$

for 各$(k, w, T)\in TRIPLE_i$ $O(|TRIPLE_i|)$

$SET(i+1)=$

$SET(i+1)\cup\{(k+c_{i+1}, w+w_{i+1}, T\cup\{i+1\})\}$

$i=i+1$

$TRIPLE_{i+1}=SET(i+1)$のうち,
実現可能かつ最小重みをもつ
3項組(m, w', T')を選択

Rap 3 (i=3)

$SET(4)=TRIPLE_3$

$SET(4)=$

$SET(4)\cup\{(35,33,\{4\})$
$(46,48,\{3,4\})$
$(58,48,\{2,4\})$
$(69,65,\{2,3,4\})$
$(68,58,\{1,4\})$
$(79,73,\{1,3,4\})$
$(91,73,\{1,2,4\})$
$(102,86,\{1,2,3,4\})\}$

$TRIPLE_4=SET(4)$
(ただし赤字含む3項組を除く)

アルゴリズム3.2.2.2(DPKP)

Step 2

for i (i=1 to n-1) $O(n\text{-}1)$

$SET(i+1)=TRIPLE_i$

for 各$(k, w, T)\in TRIPLE_i$ $O(|TRIPLE_i|)$

$SET(i+1)=SET(i+1)\cup\{(k+c_{i+1}, w+w_{i+1}, T\cup\{i+1\})\}$

$i=i+1$

$TRIPLE_{i+1}=SET(i+1)$のうち,
実現可能かつ最小重みをもつ
3項組(m, w', T')を選択

Rap 4 (i=4)

$SET(5)=TRIPLE_4$

$SET(5)=\cdots$

$TRIPLE_5=\{(0,0,\Phi)$
$\cdots$
$(69,63,\{2,3,4\})\}$

Step 3

$O(|TRIPLE_i|)$

$c:=\max\{k\in\{1,\ldots,\Sigma^n_{i=1}c_i\}|$
あるwとあるTに対して$(k, w, T)\in TRIPLE_n\}$

$c:=\max\{k\in\{1,\ldots,\Sigma^n_{i=1}c_5=113\}|$
$(k, w, T)\in TRIPLE_5\}$

出力値

$(c, w, T)\in TRIPLE_n$を満たす添字集合T

$T=\{2,3,4\}$
$(69,63,\{2,3,4\})\in TRIPLE_5$

KPの時間計算量

$$O(1) \times O(n\text{-}1) \times O(|TRIPLE_i|) + O(|TRIPLE_i|)$$

$$Time_{\mathrm{DPKP}}(I) = O(|I|^2 \cdot Max\text{-}Int(I))$$

実行時間が入力整数の個数と入力整数値の多項式

$$O(|TRIPLE_i|) \leqq \sum_{i=1}^{n} c_i \leqq n \times Max\text{-}Int(I))$$

$$n \leqq |I|$$

最大フロー問題

目的関数: sからtまでの流量

$$\max F_f = \sum_{h \in Out_G(s)} f(h) - \sum_{h \in In_G(s)} f(e)$$

制約条件: ネットワーク容量

$$f(e) \le c(e)$$

$f(e)$: 辺eのフロー関数
$c(e)$: 辺eの容量

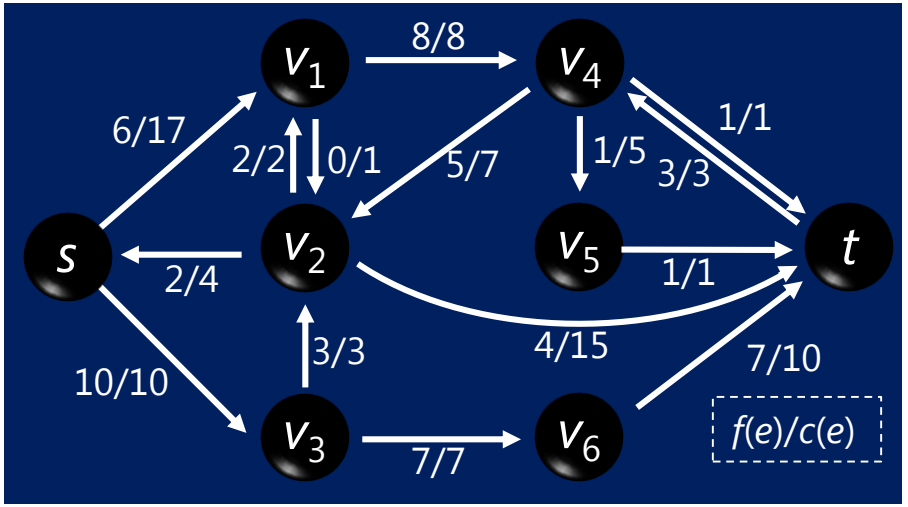
ネットワーク: $G=((V, E), c, A, s, t)$

GのフローF_f: $F_f = \sum_{h \in Out_H(s)} f(h) - \sum_{h \in In_H(s)} f(e)$

$=6+10-2$

$=14$

Gのフローはsから出ていく量で測定される

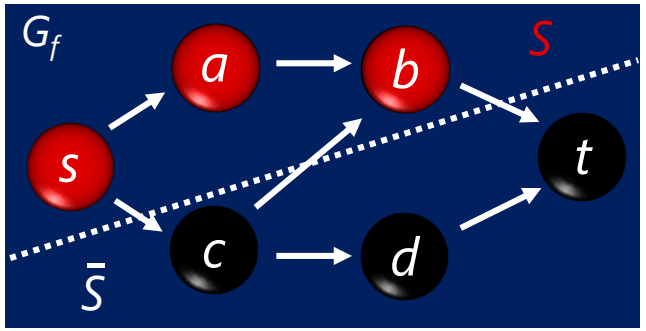


3.2.3 最大フロー問題とFord-Fulkerson法

fをネットワークGのフロー関数とする.
このとき, $s \in S$ ($S \in V-\{t\}$)なる全てのSに対して以下の等式が成り立つ.

$$F_f = \sum_{e \in E(S,\bar{S})} f(e) - \sum_{e \in E(\bar{S},S)} f(e)$$

F_f: Gのフロー



最小ネットワークカット問題は, Gに対して最小容量のGのカットを見つける問題

カットの容量 $c(S)$ $\quad c(S) = \sum_{e \in E(S,\bar{S})} c(e)$

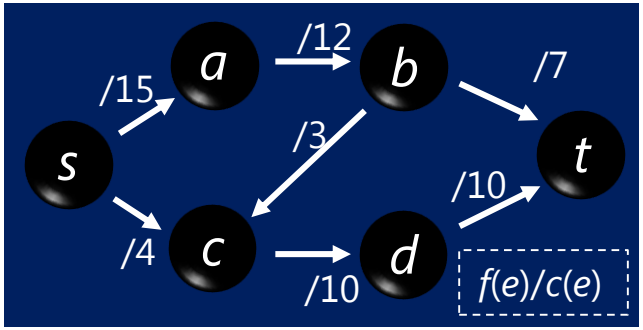
$F_f \leqq c(S)$より, 最大フロー問題は, 最小カット問題と双対

3.2.3 最大フロー問題とFord-Fulkerson法

アルゴリズム3.2.3.10(Ford-Fulkersonアルゴリズム)

入力値

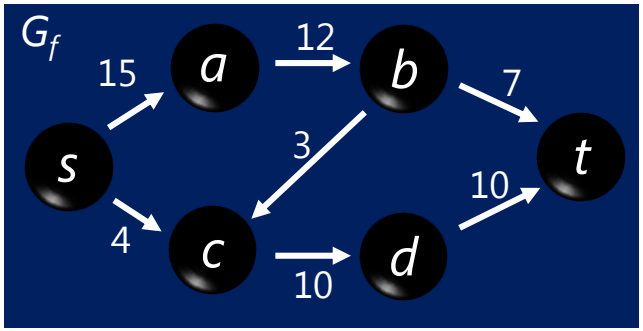
ネットワーク: $G=((V, E), c, A, s, t)$の(V, E, c, s, t)



Step 1

$O(|E|)$

Gの初期フロー関数fの決定
-例) 任意の辺$e \in E$に対して$f(e)=0$



G_f: 残留ネットワーク

3.2.3 最大フロー問題とFord-Fulkerson法

アルゴリズム3.2.3.10(Ford-Fulkersonアルゴリズム)

Rap 1

$O(|1|)$ **Step 2** S: $=\{s\}$, $\bar{S}$: $=V-S$

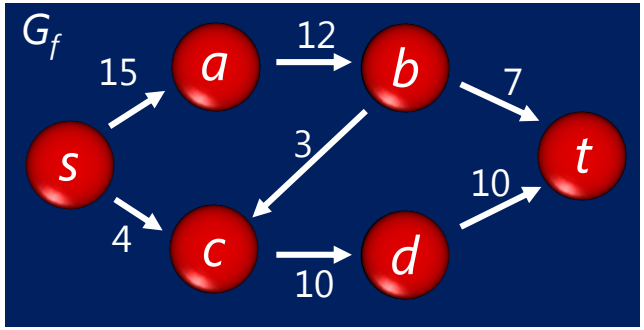
Step 3

次の条件を満たす
$e=(u, v)\in E(S, \bar{S})\cup E(\bar{S}, S)$をみつける

$O(|E|)$

条件	$c(e)-f(e)>0$	①: e$\in E(S, \bar{S})$の場合
	$f(e)>0$	②: e$\in E(\bar{S}, S)$の場合

・条件を満たす辺eが存在: $S=S\cup\{v\}$ ①の場合
$S=S\cup\{u\}$ ②の場合



G_f: 残留ネットワーク

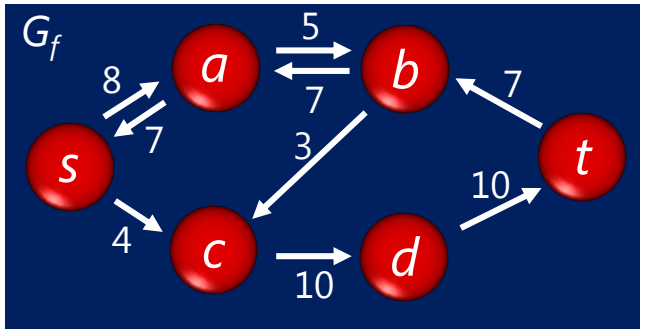
Step 4

P: s,a,b,t $res(P)=7$

Sの頂点のみからなるsからtへの増加路Pをみつける

$O(|V|)$

$res(P)$を計算: $res(P)=\min\{res(e_i)|i=0,\ldots,k\}$
$f'(e) = f(e) + res(P)$



Step 2 に戻る

アルゴリズム3.2.3.10(Ford-Fulkersonアルゴリズム)

Rap 2

$O(|1|)$ **Step 2** S: $=\{s\}$, $\bar{S}$: $=V-S$

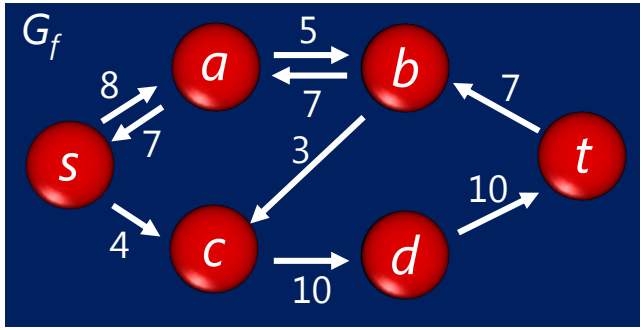
Step 3

次の条件を満たす
$e=(u, v)\in E(S, \bar{S})\cup E(\bar{S}, S)$をみつける

$O(|E|)$

条件	$c(e)-f(e)>0$	①: e$\in E(S, \bar{S})$の場合
	$f(e)>0$	②: e$\in E(\bar{S}, S)$の場合

・条件を満たす辺eが存在: $S=S\cup\{v\}$ ①の場合
$S=S\cup\{u\}$ ②の場合



G_f: 残留ネットワーク

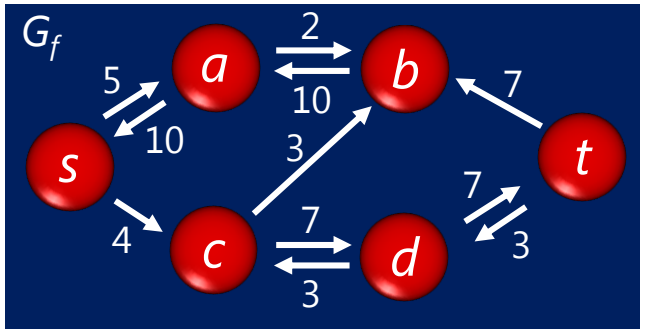
Step 4

P: s,a,b,c,d,t $res(P)=3$

Sの頂点のみからなるsからtへの増加路Pをみつける

$O(|V|)$

$res(P)$を計算: $res(P)=\min\{res(e_i)|i=0,\ldots,k\}$
$f'(e) = f(e) + res(P)$



Step 2 に戻る

アルゴリズム3.2.3.10(Ford-Fulkersonアルゴリズム)

Rap 3

$O(|1|)$ **Step 2** S: $=\{s\}$, $\bar{S}$: $=V-S$

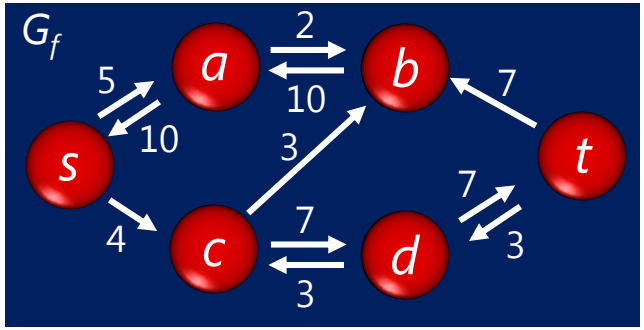
Step 3

次の条件を満たす
$e=(u, v)\in E(S, \bar{S})\cup E(\bar{S}, S)$をみつける

$O(|E|)$

条件	$c(e)-f(e)>0$	①: e$\in E(S, \bar{S})$の場合
	$f(e)>0$	②: e$\in E(\bar{S}, S)$の場合

・条件を満たす辺eが存在: $S=S\cup\{v\}$ ①の場合
$S=S\cup\{u\}$ ②の場合



G_f: 残留ネットワーク

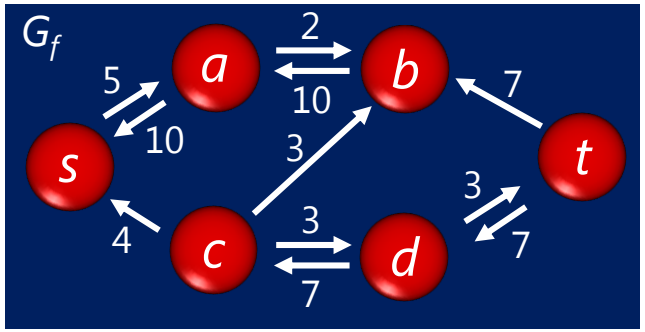
Step 4

Sの頂点のみからなるsからtへの増加路Pをみつける

$O(|V|)$

$res(P)$を計算: $res(P)=\min\{res(e_i)|i=0,\ldots,k\}$
$f'(e) = f(e) + res(P)$

P: s,c,d,t $res(P)=4$



Step 2 に戻る

3.2.3 最大フロー問題とFord-Fulkerson法

アルゴリズム3.2.3.10(Ford-Fulkersonアルゴリズム)

Rap 4

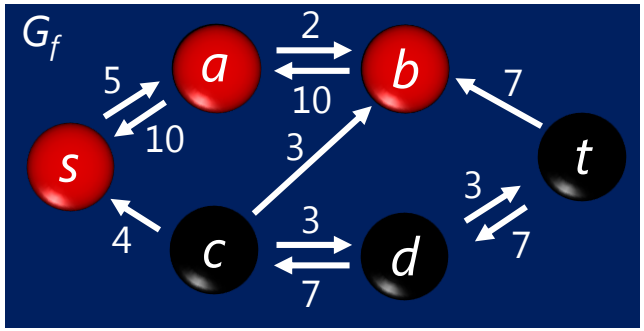
$O(|1|)$ **Step 2** S: $=\{s\}$, $\bar{S}$: $=V-S$

Step 3 次の条件を満たす
$e=(u, v)\in E(S, \bar{S})\cup E(\bar{S}, S)$をみつける

$O(|E|)$

条件		
	$c(e)-f(e)>0$	①: e$\in E(S, \bar{S})$の場合
	$f(e)>0$	②: e$\in E(\bar{S}, S)$の場合

・条件を満たす辺eが存在: $S=S\cup\{v\}$ ①の場合
$S=S\cup\{u\}$ ②の場合



G_f: 残留ネットワーク

S拡大不可の場合, 最適解に到達

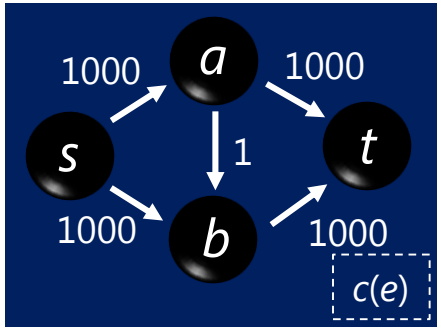
Ford-Fulkersonアルゴリズムの時間計算量

繰り返し回数: 高々$F_f = c(S) < \sum_{e \in E} c(e) < |E| \times Max\text{-}Int(G)$

$$Time_{\text{Ford-Fulkerson}}(I) = O(|E|^2 \times Max\text{-}Int(G))$$

$s \to a \to t$
$s \to b \to t$
ループの繰り返し回数は1回

$s \to a \to b \to t$
$s \to b \to a \to t$
ループの繰り返し回数は1,000回



3.4.1 基本概念

分枝限定法

アプローチ①と③に基づく考え方

付加的な情報によって, 効率的に探索

最適化問題を解くためのアルゴリズム設計

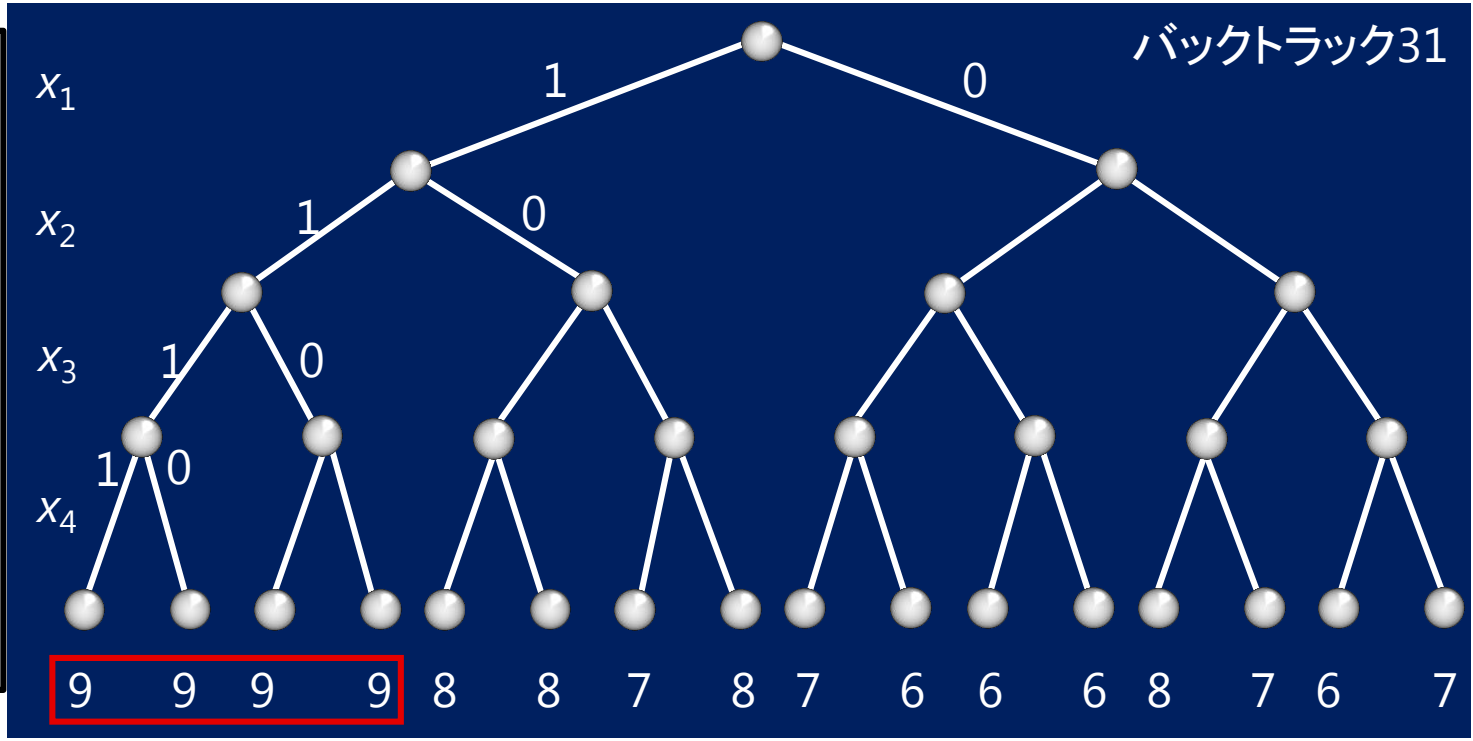
しらみつぶしの中で, 最適解が含まれないとわかった枝を省略

3.4.2 MAX-SATとTSPに対する適用例

MAX-SAT

充足性最大化問題

例) $\Phi(x_1, x_2, x_3, x_4)=$
$\wedge(x_1 \vee \bar{x}_2)$
$\wedge(x_1 \vee x_3 \vee \bar{x}_4)$
$\wedge(\bar{x}_1 \vee x_2)$
$\wedge(x_1 \vee \bar{x}_3 \vee x_4)$
$\wedge(x_2 \vee x_3 \vee \bar{x}_4)$
$\wedge(x_1 \vee \bar{x}_3 \vee \bar{x}_4)$
$\wedge(x_3)$
$\wedge(x_1 \vee x_4)$
$\wedge(\bar{x}_1 \vee \bar{x}_3)$ ← 節
$\wedge(x_1)$

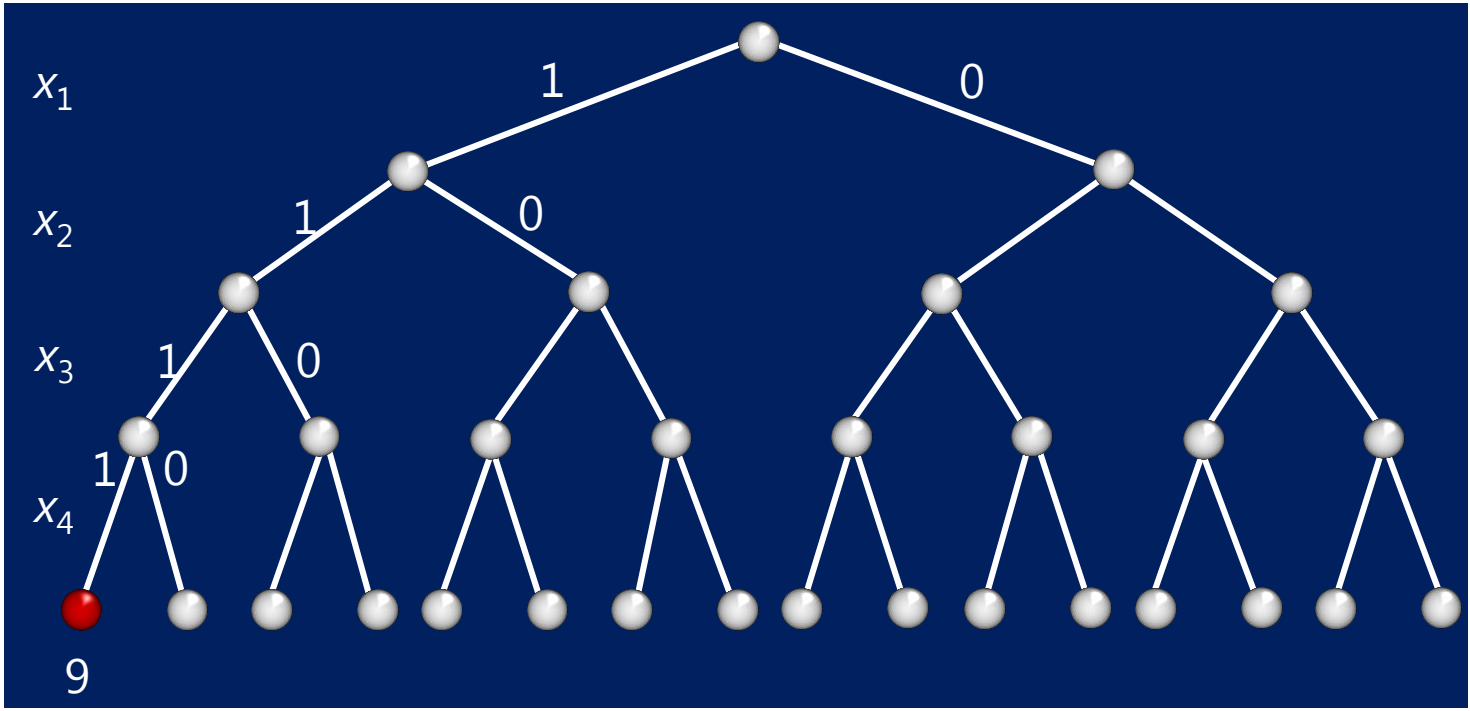


→最適解

完全な探索木(しらみつぶし)

3.4.2 MAX-SATとTSPに対する適用例

例) $\Phi(x_1, x_2, x_3, x_4)=$
$\wedge(x_1 \vee \bar{x}_2)$
$\wedge(x_1 \vee x_3 \vee \bar{x}_4)$
$\wedge(\bar{x}_1 \vee x_2)$
$\wedge(x_1 \vee \bar{x}_3 \vee x_4)$
$\wedge(x_2 \vee x_3 \vee \bar{x}_4)$
$\wedge(x_1 \vee \bar{x}_3 \vee \bar{x}_4)$
$\wedge(x_3)$
$\wedge(x_1 \vee x_4)$
$\wedge(\bar{x}_1 \vee \bar{x}_3)$
$\wedge(x_1)$

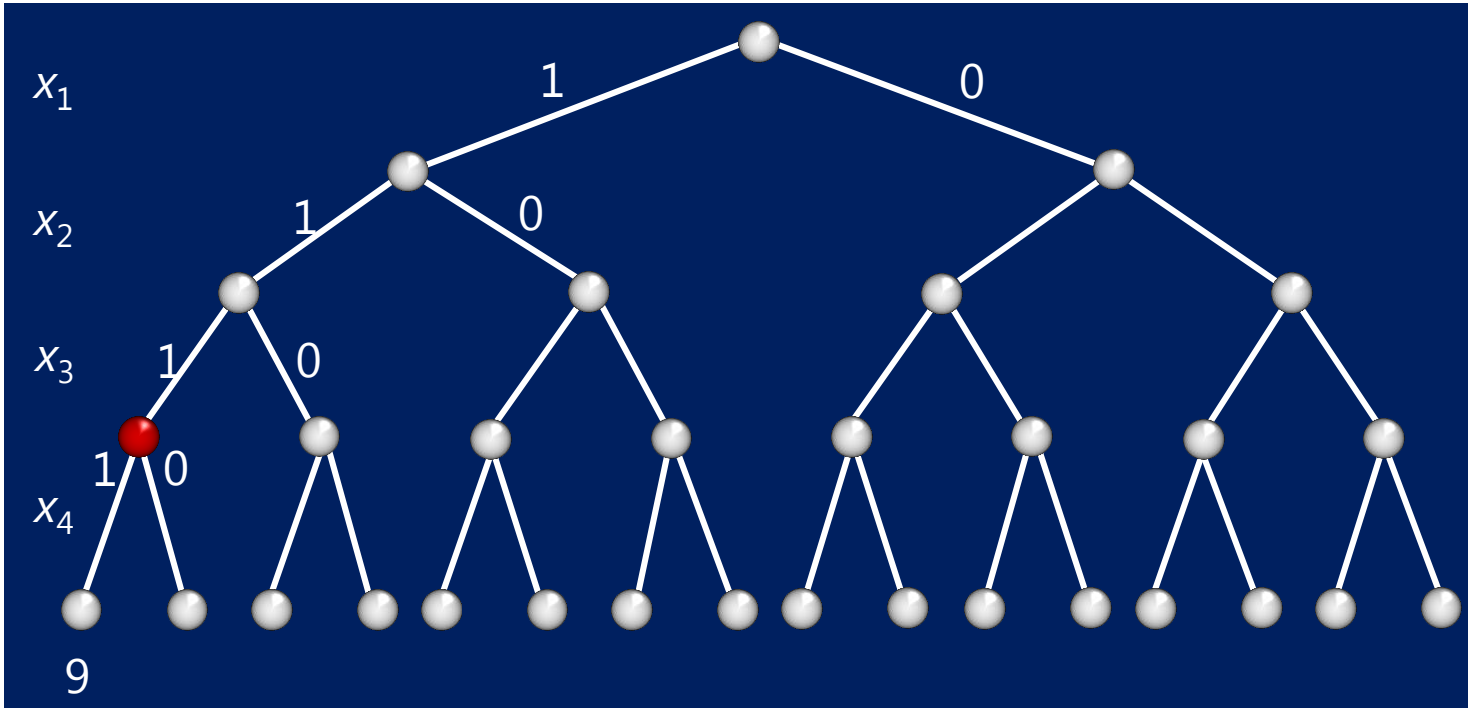


→充足されていない(すなわち, 10じゃない)
→x_4を0にすれば充足されるか?
→節$(\bar{x}_1 \vee \bar{x}_3)$が充足されない
→部分木を切断

深さ優先探索による分枝限定法

3.4.2 MAX-SATとTSPに対する適用例

例) $\Phi(x_1, x_2, x_3, x_4)=$
$\wedge(x_1 \vee \bar{x}_2)$
$\wedge(x_1 \vee x_3 \vee \bar{x}_4)$
$\wedge(\bar{x}_1 \vee x_2)$
$\wedge(x_1 \vee \bar{x}_3 \vee x_4)$
$\wedge(x_2 \vee x_3 \vee \bar{x}_4)$
$\wedge(x_1 \vee \bar{x}_3 \vee \bar{x}_4)$
$\wedge(x_3)$
$\wedge(x_1 \vee x_4)$
$\wedge(\bar{x}_1 \vee \bar{x}_3)$
$\wedge(x_1)$

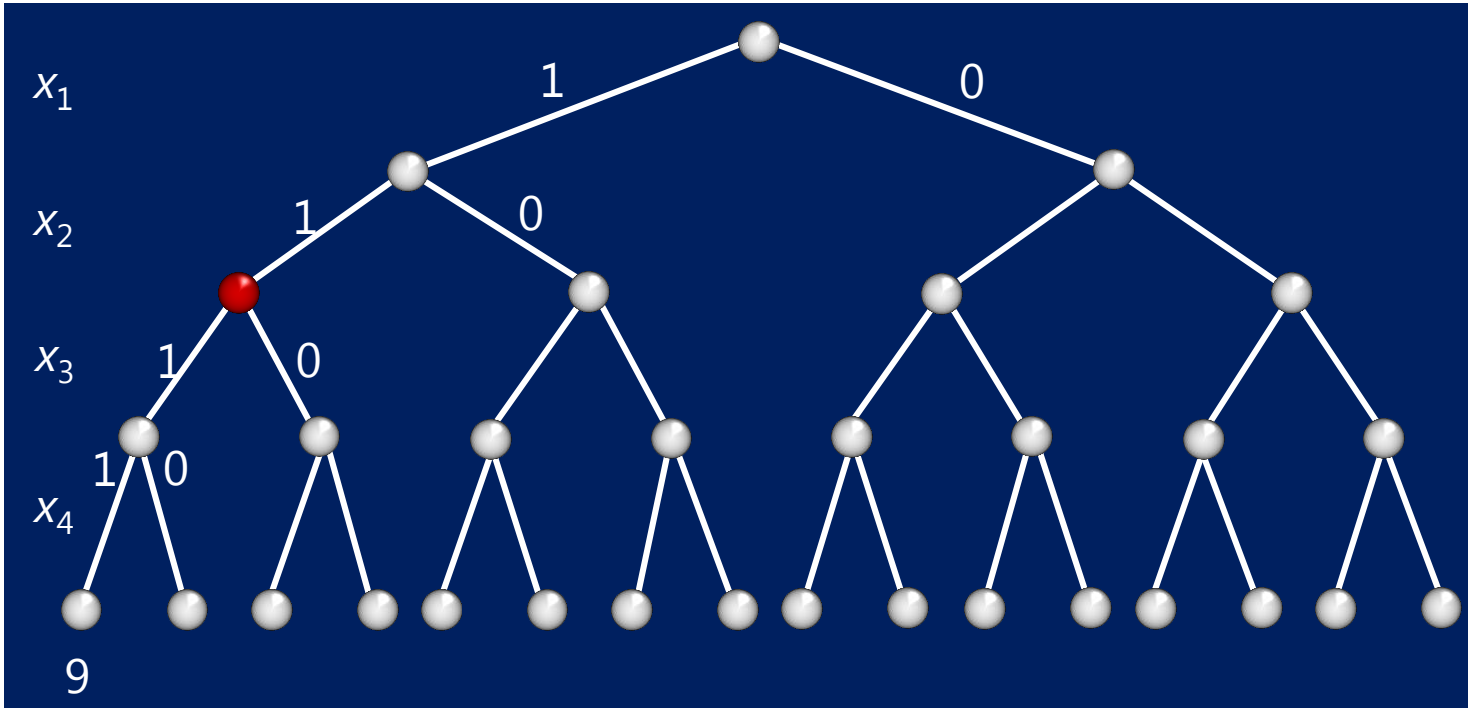


→充足されていない(すなわち, 10じゃない)
→x_4を0にすれば充足されるか?
→節$(\bar{x}_1 \vee \bar{x}_3)$が充足されない
→部分木を切断

深さ優先探索による分枝限定法

3.4.2 MAX-SATとTSPに対する適用例

例) $\Phi(x_1, x_2, x_3, x_4)=$
$\wedge(x_1 \vee \bar{x}_2)$
$\wedge(x_1 \vee x_3 \vee \bar{x}_4)$
$\wedge(\bar{x}_1 \vee x_2)$
$\wedge(x_1 \vee \bar{x}_3 \vee x_4)$
$\wedge(x_2 \vee x_3 \vee \bar{x}_4)$
$\wedge(x_1 \vee \bar{x}_3 \vee \bar{x}_4)$
$\wedge(x_3)$
$\wedge(x_1 \vee x_4)$
$\wedge(\bar{x}_1 \vee \bar{x}_3)$
$\wedge(x_1)$

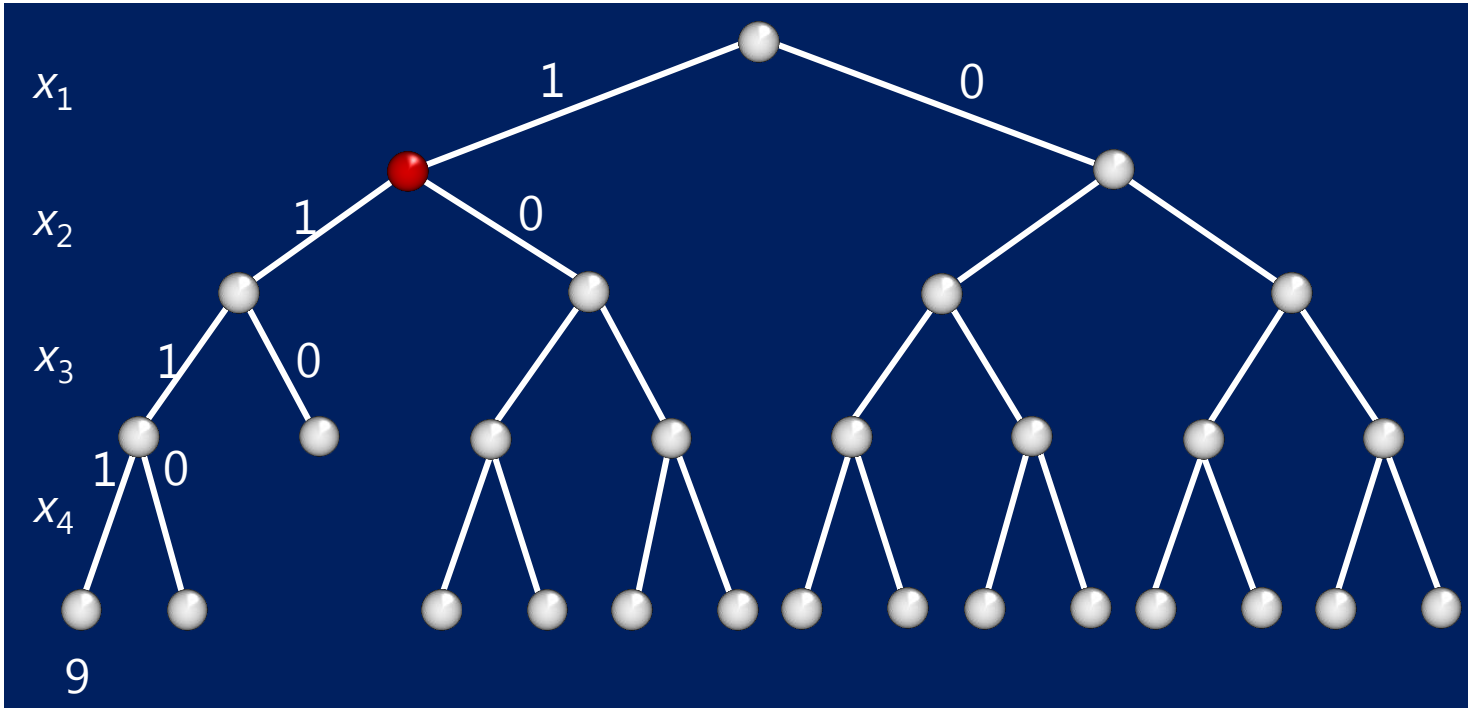


→充足されていない(すなわち, 10じゃない)
→x_3を0にすれば充足されるか?
→節(x_3)が充足されない
→部分木を切断

深さ優先探索による分枝限定法

3.4.2 MAX-SATとTSPに対する適用例

例) $\Phi(x_1, x_2, x_3, x_4)=$
$\wedge(x_1 \vee \bar{x}_2)$
$\wedge(x_1 \vee x_3 \vee \bar{x}_4)$
$\wedge(\bar{x}_1 \vee x_2)$
$\wedge(x_1 \vee \bar{x}_3 \vee x_4)$
$\wedge(x_2 \vee x_3 \vee \bar{x}_4)$
$\wedge(x_1 \vee \bar{x}_3 \vee \bar{x}_4)$
$\wedge(x_3)$
$\wedge(x_1 \vee x_4)$
$\wedge(\bar{x}_1 \vee \bar{x}_3)$
$\wedge(x_1)$

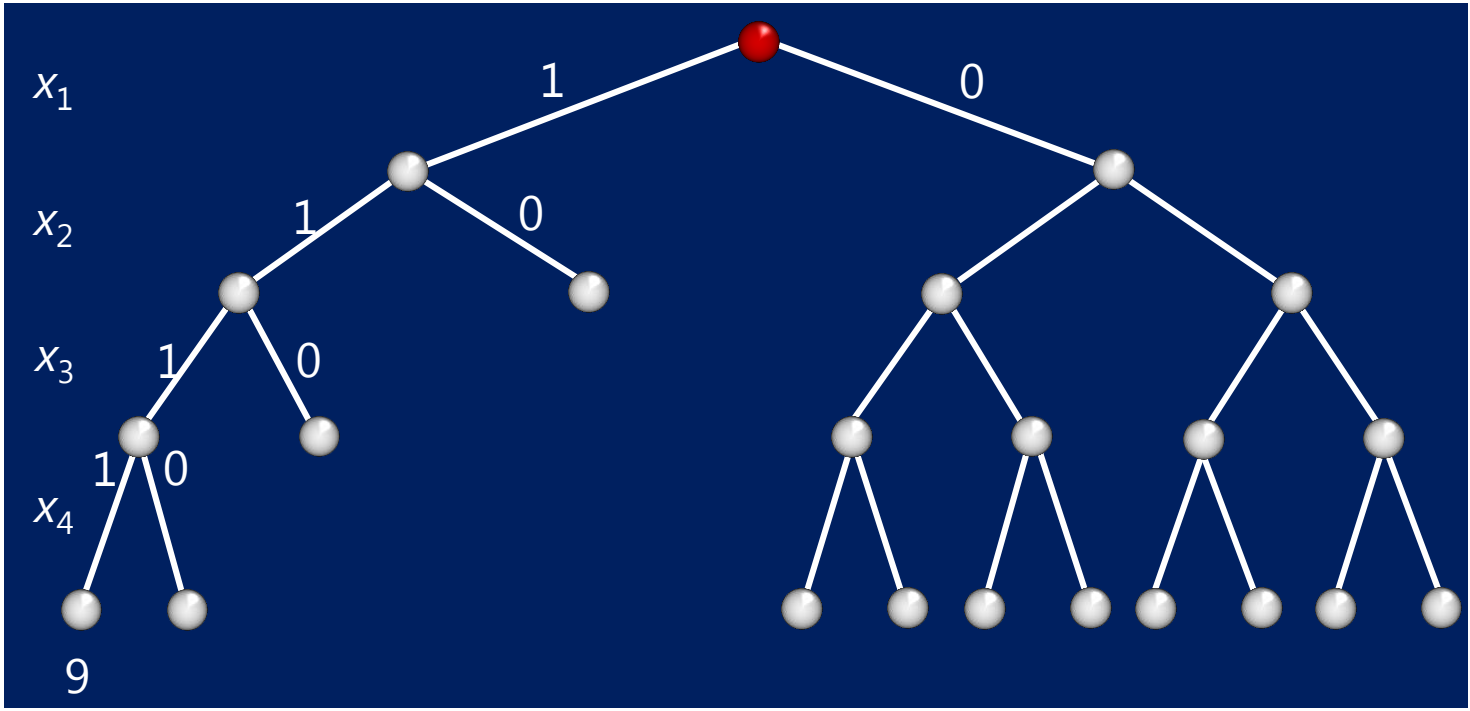


→充足されていない(すなわち, 10じゃない)
→x_2を0にすれば充足されるか?
→節$(\bar{x}_1 \vee x_2)$が充足されない
→部分木を切断

深さ優先探索による分枝限定法

3.4.2 MAX-SATとTSPに対する適用例

例) $\Phi(x_1, x_2, x_3, x_4)=$
$\wedge(x_1 \vee \bar{x}_2)$
$\wedge(x_1 \vee x_3 \vee \bar{x}_4)$
$\wedge(\bar{x}_1 \vee x_2)$
$\wedge(x_1 \vee \bar{x}_3 \vee x_4)$
$\wedge(x_2 \vee x_3 \vee \bar{x}_4)$
$\wedge(x_1 \vee \bar{x}_3 \vee \bar{x}_4)$
$\wedge(x_3)$
$\wedge(x_1 \vee x_4)$
$\wedge(\bar{x}_1 \vee \bar{x}_3)$
$\wedge(x_1)$

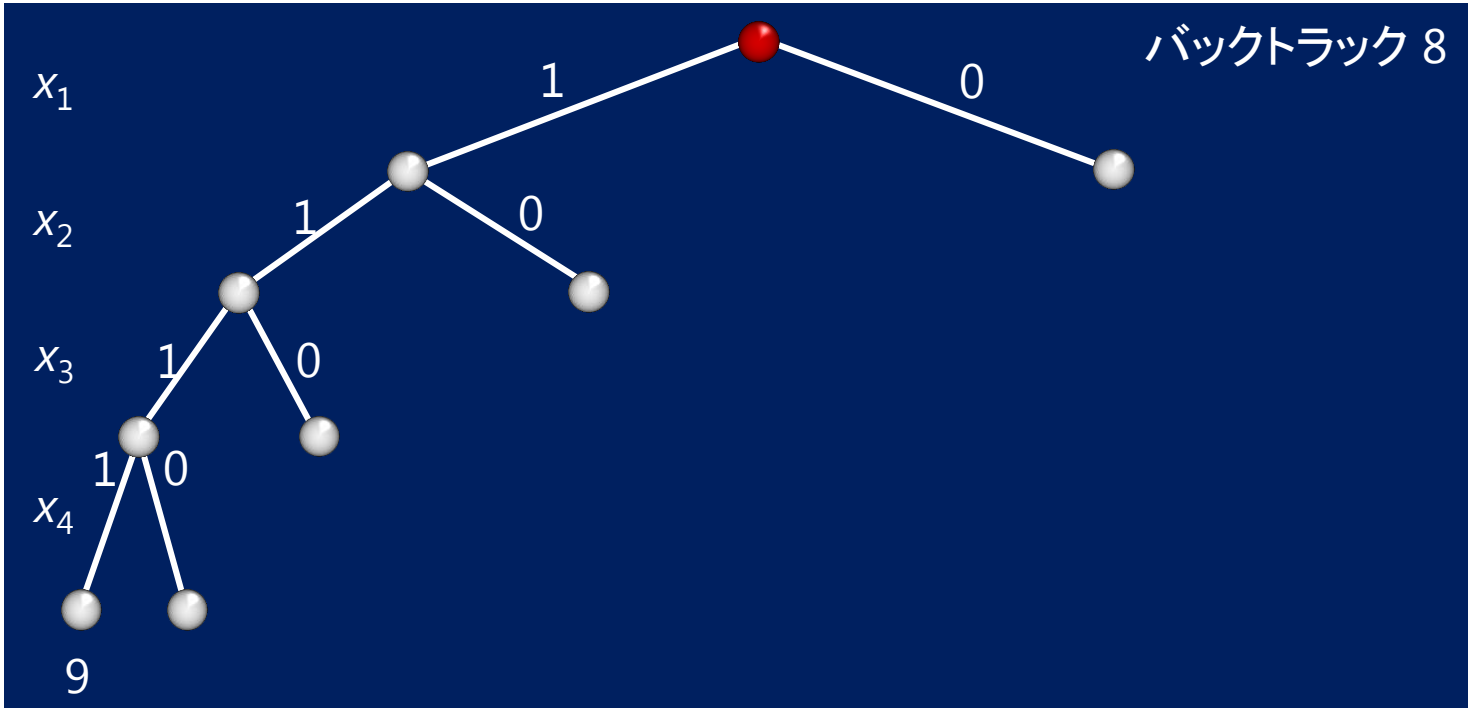


→充足されていない(すなわち, 10じゃない)
→x_1を0にすれば充足されるか?
→節(x_1)が充足されない
→部分木を切断

深さ優先探索による分枝限定法

3.4.2 MAX-SATとTSPに対する適用例

例) $\Phi(x_1, x_2, x_3, x_4)=$
$\wedge(x_1 \vee \bar{x}_2)$
$\wedge(x_1 \vee x_3 \vee \bar{x}_4)$
$\wedge(\bar{x}_1 \vee x_2)$
$\wedge(x_1 \vee \bar{x}_3 \vee x_4)$
$\wedge(x_2 \vee x_3 \vee \bar{x}_4)$
$\wedge(x_1 \vee \bar{x}_3 \vee \bar{x}_4)$
$\wedge(x_3)$
$\wedge(x_1 \vee x_4)$
$\wedge(\bar{x}_1 \vee \bar{x}_3)$
$\wedge(x_1)$

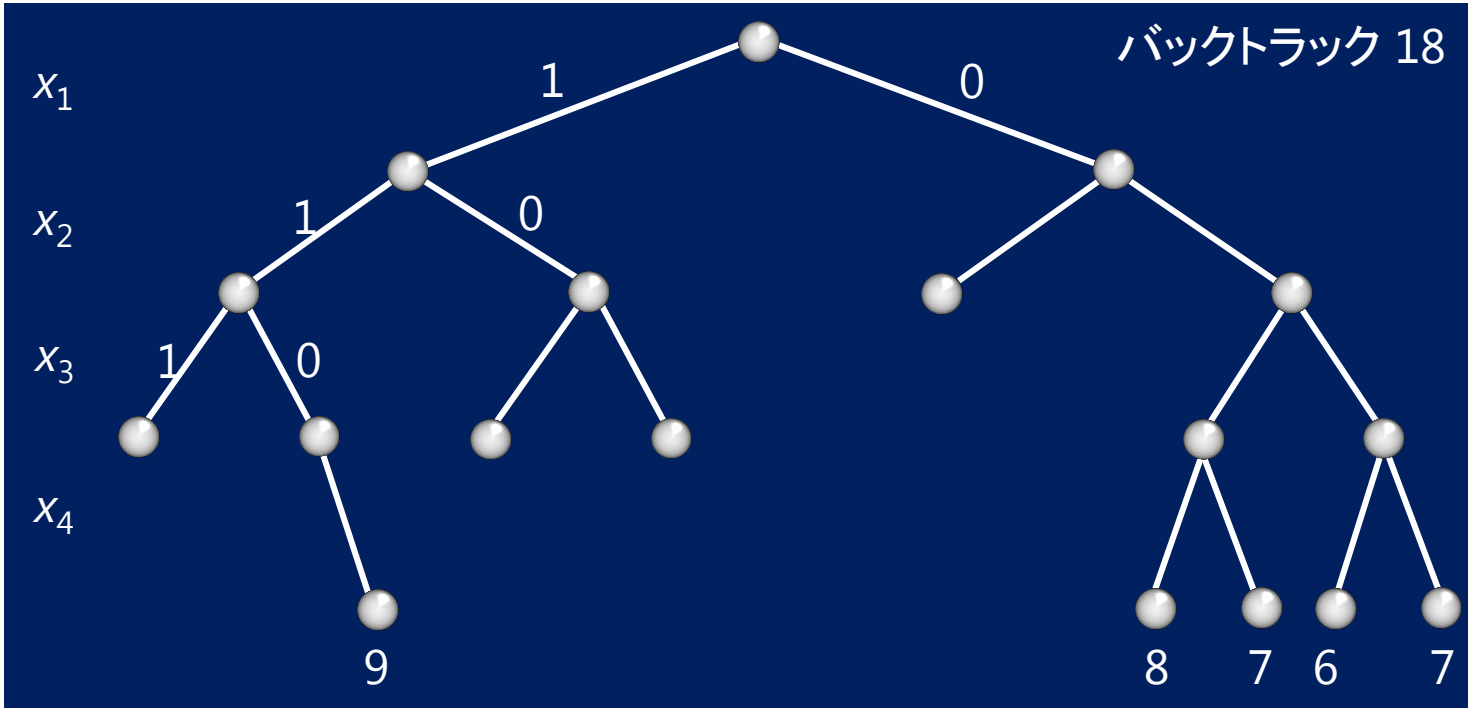


最大値9

深さ優先探索による分枝限定法

3.4.2 MAX-SATとTSPに対する適用例

例) $\Phi(x_1, x_2, x_3, x_4)=$
$\wedge(x_1 \vee \bar{x}_2)$
$\wedge(x_1 \vee x_3 \vee \bar{x}_4)$
$\wedge(\bar{x}_1 \vee x_2)$
$\wedge(x_1 \vee \bar{x}_3 \vee x_4)$
$\wedge(x_2 \vee x_3 \vee \bar{x}_4)$
$\wedge(x_1 \vee \bar{x}_3 \vee \bar{x}_4)$
$\wedge(x_3)$
$\wedge(x_1 \vee x_4)$
$\wedge(\bar{x}_1 \vee \bar{x}_3)$
$\wedge(x_1)$



探索方法によって計算回数が大きく異なる

深さ優先探索による分枝限定法(x_i=0から探索)

3.4.2 MAX-SATとTSPに対する適用例

TSP

目的関数: 総移動コスト

制約条件:

全ての頂点で,
どれか1本の入路, 出路が使われる

一筆書きで全頂点を巡回

$$\max z_{\mathrm{TSP}} \sum c_{ij} x_{ij}$$

$$\sum_{(i,j)\in\delta_i^+} x_{ij} = 1, \quad \sum_{(i,j)\in\delta_j^-} x_{ij} = 1$$

$$x_{ij} \in \{0,1\}, (i,j) \in E$$

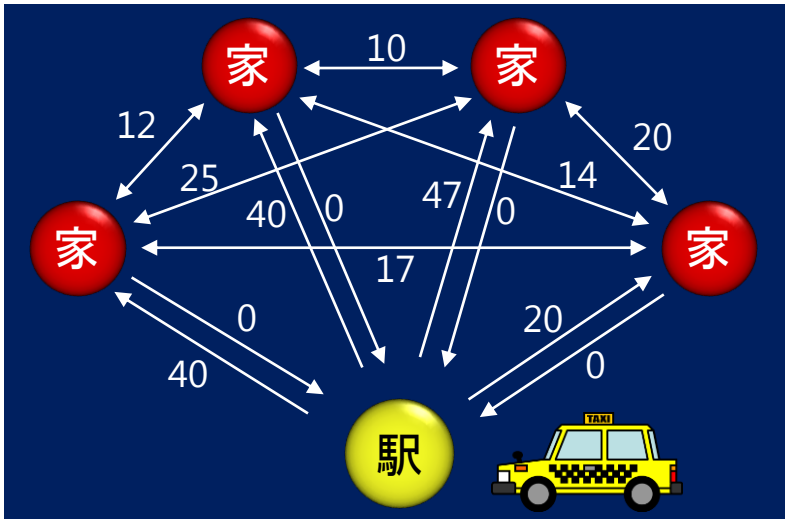
$$X = \{(i,j) \in E \mid x_{ij} = 1\}$$

が部分巡回路を含まない

例題

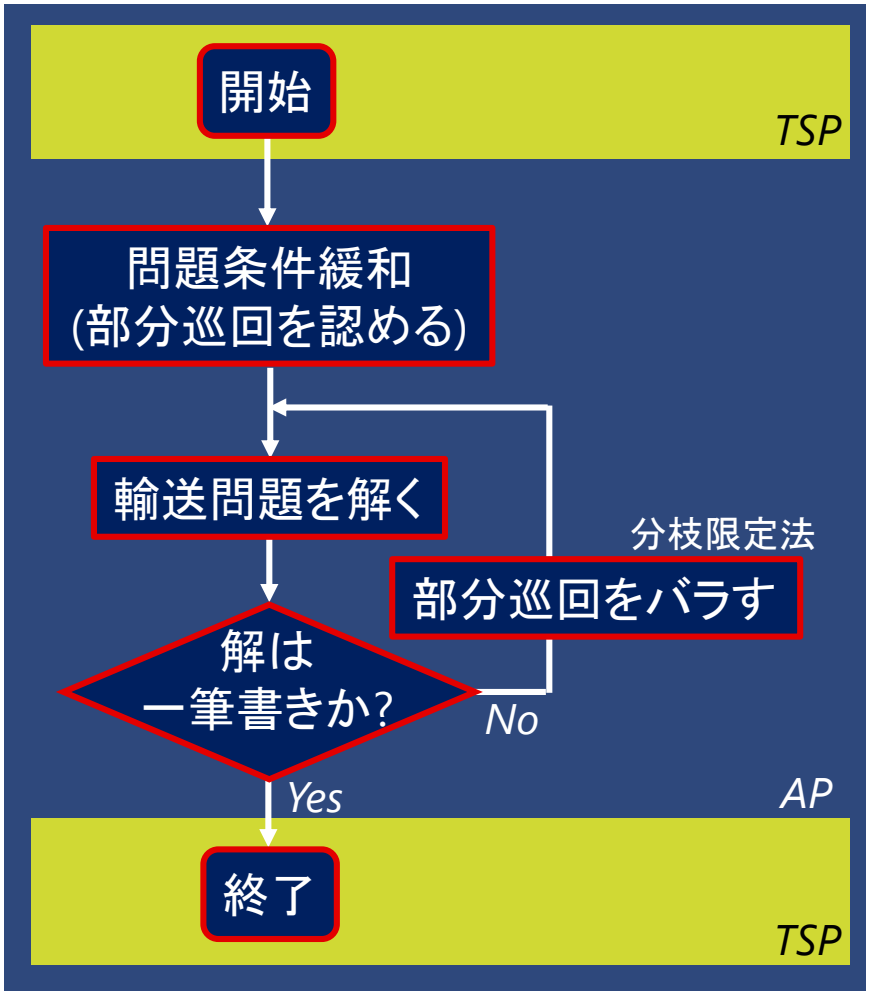
タクシー: 駅から4つの家を廻り, 駅に戻る

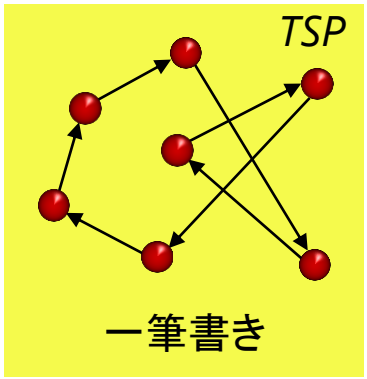
・緩和問題 (*AP*)
・分枝限定法
　　2つを組み合わせて*TSP*を解く

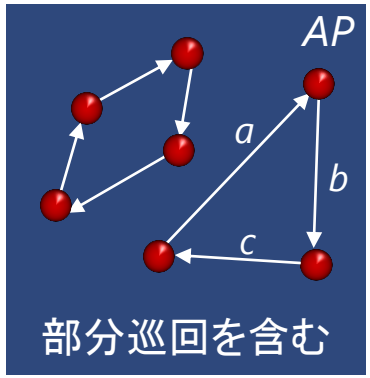


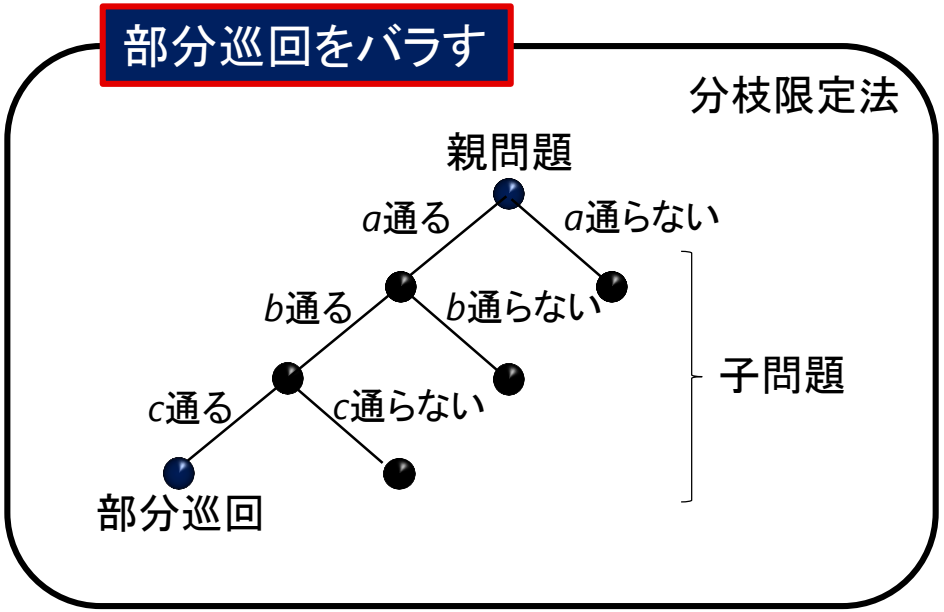
http://www.ocw.titech.ac.jp/index.php?module=General&action=DownLoad&file=200920464-5-0-39.pdf&type=cal&JWC=200920464

3.4.2 MAX-SATとTSPに対する適用例









3.4.2 MAX-SATとTSPに対する適用例

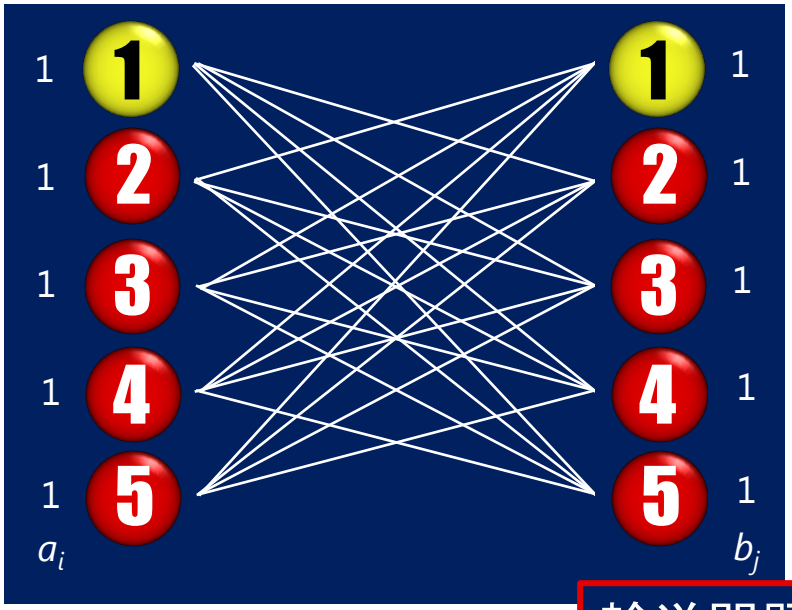
親問題を解く

AP: *TSP*から部分巡回路除去制約式を除いた問題

⇕

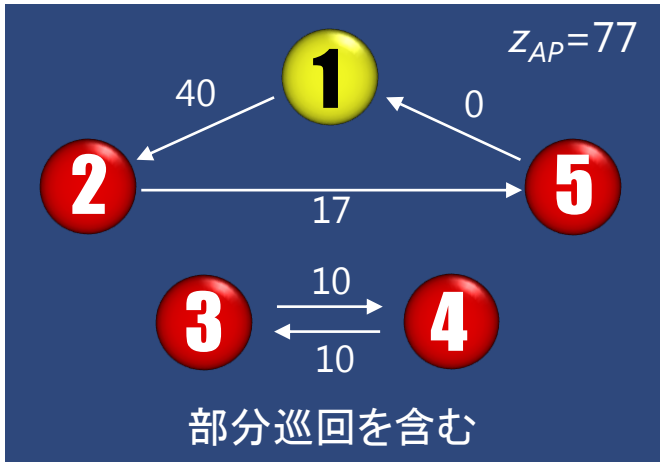
輸送問題

・供給地と需要地の個数が等しい
・すべての供給量a_i需要量とb_jがともに1



$O((n\text{-}1)\times(n\text{-}1)!)$ 輸送問題

輸送問題の解法 (飛び石法) を適用した結果



総コスト(目的関数) $z_{AP}=77 \leqq z_{TSP}$

3.4.2 MAX-SATとTSPに対する適用例

子問題を解く

Step 1 ネットワーク中に以下のリンクを設定

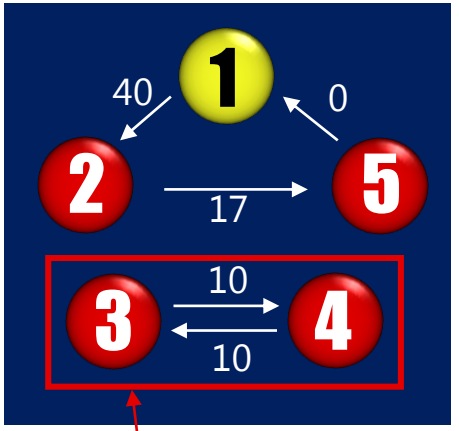
- **通ってはいけないリンク** E_0
- **必ず通るリンク** E_1

Step 2 分枝操作: 親問題をs個の子問題に分割

部分巡回路のうち, 枝数が最小の部分巡回路に着目 (s: 枝数)

親問題 $(AP[E_0, E_1]) = (AP[\{\Phi\}, \{\Phi\}])$ **子問題** $(AP[E_0, E_1])$

子問題1: $(AP[E_0 \cup \{(1,2)\}, E_1])$
子問題2: $(AP[E_0 \cup \{(2,3)\}, E_1 \cup \{(1,2)\}])$
⋮
子問題s: $(AP[E_0 \cup \{(s,1)\}, E_1 \cup \{(1,2), (2,3),\ldots,(s-1,s)\}])$



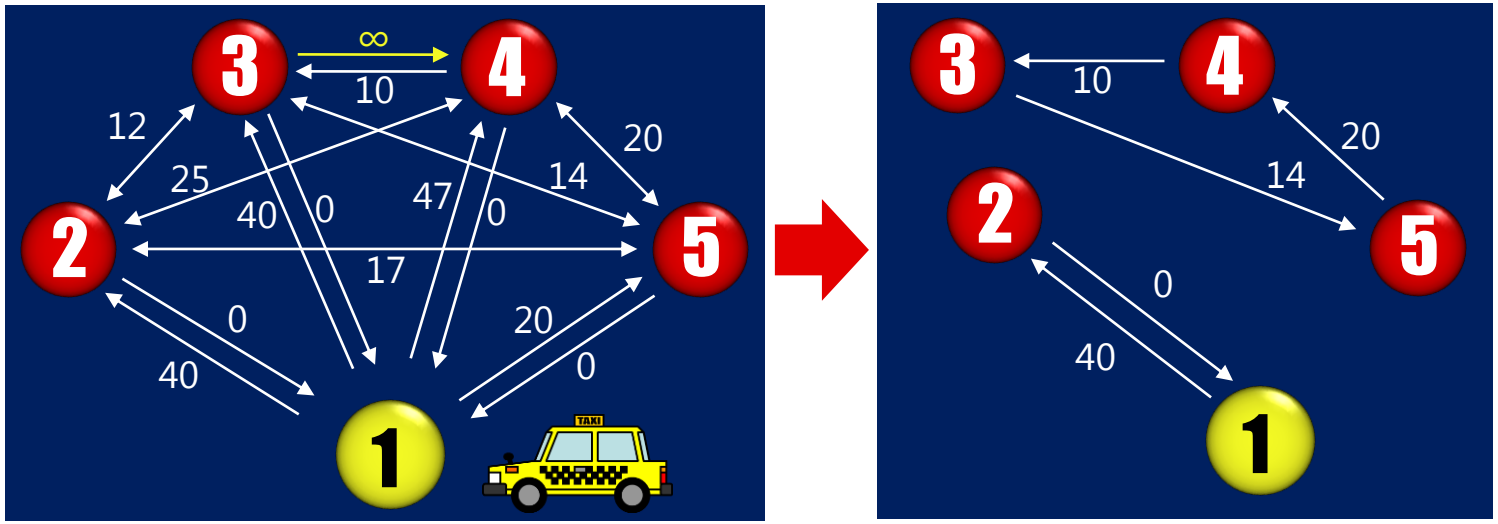
着目(s=2)

子問題1:
$(AP[\{(3,4)\}, \{\Phi\}])$

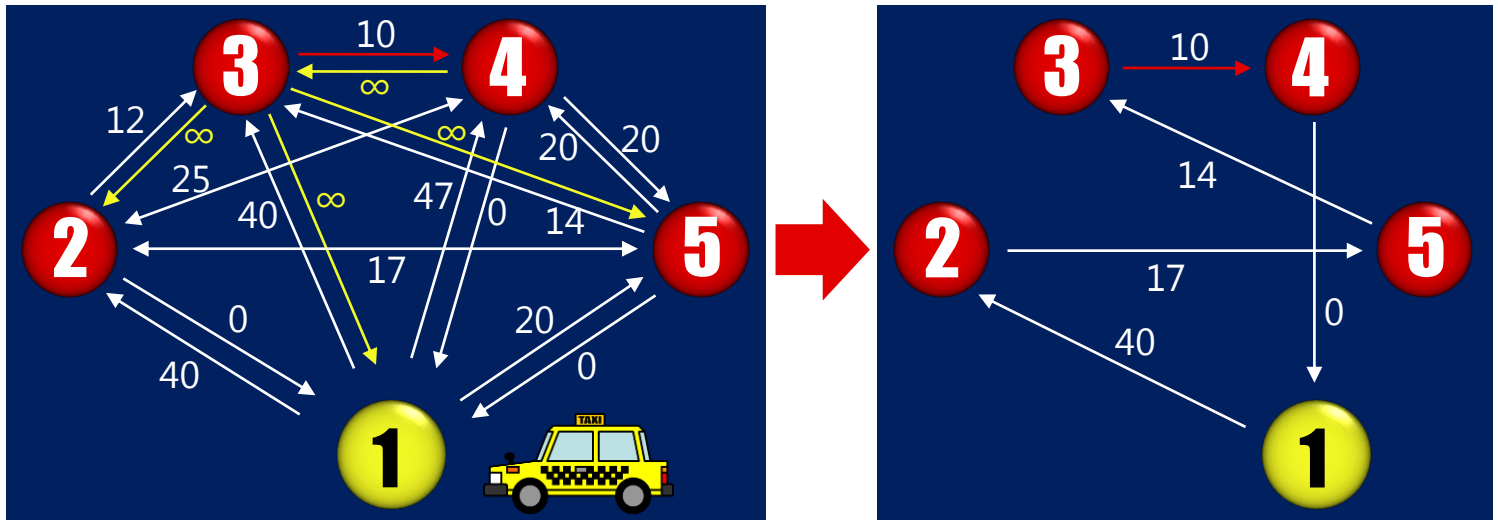
子問題2:
$(AP[\{(4,3)\}, \{(3,4)\}])$

3.4.2 MAX-SATとTSPに対する適用例

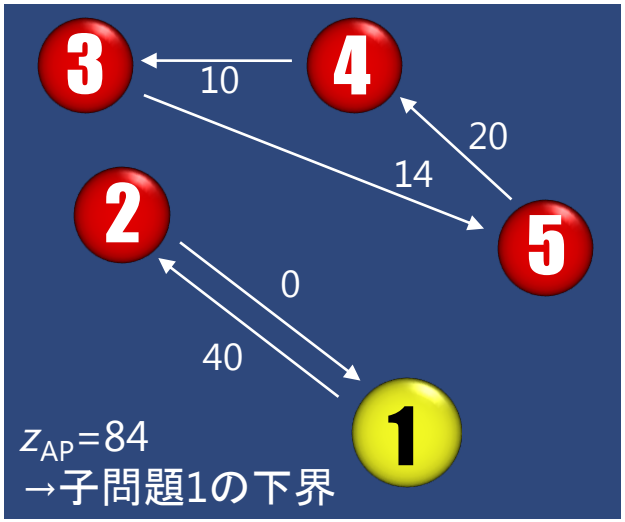
子問題1:
(*TSP*[{(3,4)}, {*Φ*}])

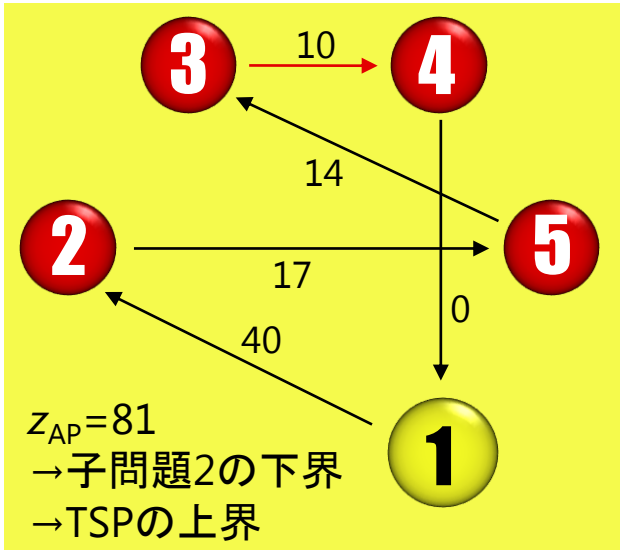


子問題2:
(*TSP*[{(4,3)}, {(3,4)}])



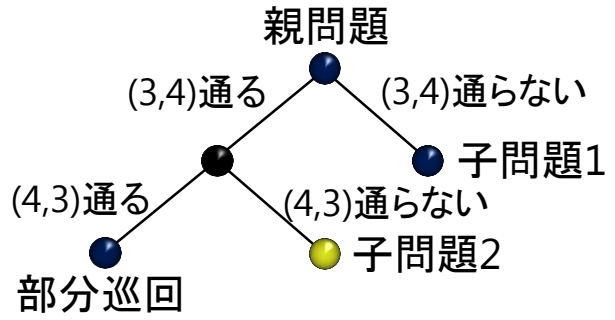
3.4.2 MAX-SATとTSPに対する適用例





部分巡回をバラす

分枝限定法



$z_{AP親} \geqq 77$

$z_{AP子1} \geqq 84$

$z_{AP子2} \geqq 81 \geqq z_{TSP}$

3.4.2 MAX-SATとTSPに対する適用例

分枝限定法の時間計算量

以下の要因によって決定

・木$T_{M(x)}$の探索戦略
・バックトラッキングによる$T_{M(x)}$の構成の種類

$T_{M(x)}$: 根付き木

深さ優先探索を用いるべきケース

•前通りのパターンを列挙し、結果をまとめる必要がある場合
•文字列などを探索するときに、辞書順であることが求められる場合

幅優先探索を用いるべきケース

•始点から最も近いものを求めたいケース
•探索範囲は広いが、ある程度近くに求めたい解が存在することがわかっているケース
•探索範囲が広く、深さ優先探索ではスタックが大量に使われてしまうケース

3.4 分枝限定法のまとめ

分枝限定法アルゴリズムは次の2つからなる

1. 最適コストの限界を計算

2. バックトラック木からある部分木を切断するために,
   1で計算した限界を利用

まとめ

①困難問題の集合を, 易しい入力インスタンスと困難な入力インスタンスに分割

②増加率が非常に遅い最悪指数時間計算量のアルゴリズムを設計

③問題の条件緩和

擬多項式時間アルゴリズム

分枝限定法