Efficient Bandit Combinatorial Optimization Algorithm with Zero-suppressed Binary Decision Diagrams

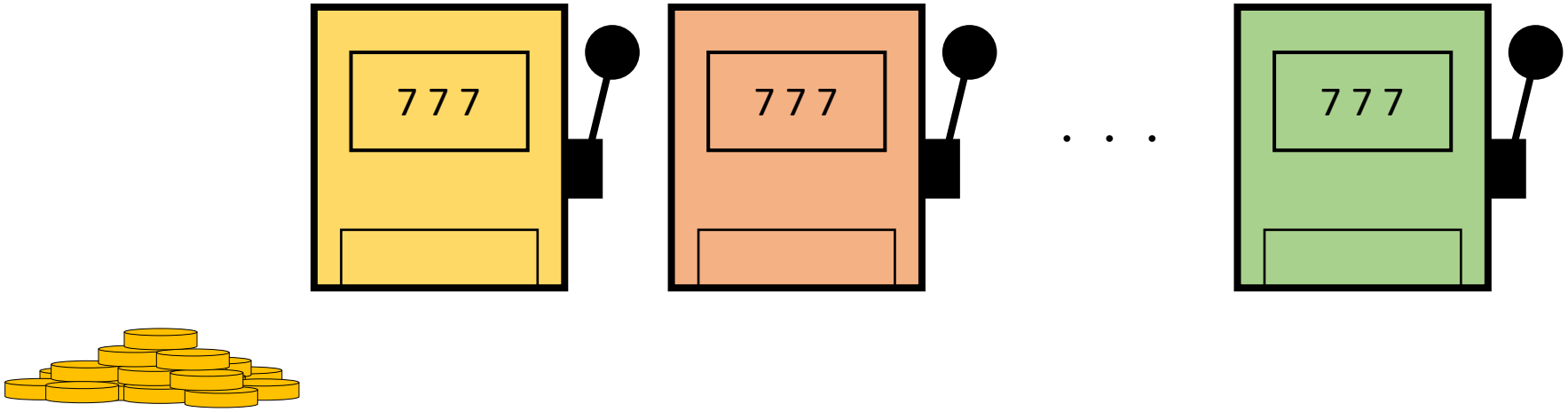
Shinsaku Sakaue, Masakazu Ishihata, Shin-ichi Minato

Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics, PMLR 84:585-594, 2018.

2021 6/8

小川 瑞貴

バンディット問題とは？



- 手元にはある枚数のコインがある
- 複数のスロットマシンがあり，それぞれの報酬の確率はわからない（未知の確率分布）

手持ちにあるコインで、報酬を最大化させたい！
→一部のコインでどのマシンが当たりやすいかを**探索**
マシンの報酬の当たりやすさを把握
→当たりやすいマシンにかける**知識利用**
探索と知識利用はトレードオフの関係

バンディット問題とは？

「選択肢集合の中から1つの要素を選択し、その選択肢に対する報酬を得るが、それ以外の選択肢の報酬情報は得られない」
というプロセスを繰り返し行い、報酬和の最大化を目指す逐次選択問題

利用例
・インターネット広告配信
・推薦システム
・ゲーム木探索
・オンライン経路制御
・ルーティング問題
・・・などがある

■確率的バンディット問題
各アームの報酬が何らかの確率分布に従って生成される

■敵対的バンディット問題
敵対者が存在し、プレイヤーの選択に応じて、報酬を決定する

問題設定

・プレイヤーvs敵対者のTラウンドからなる逐次選択問題

・プレイヤーは選択肢集合Sの中からスーパーアームX_tを選択

$X_t = \{i, \dots\} \in E$

・敵対者が設定した損失ベクトルl_tからのコストを

プレイヤーは受け取る

目標

プレイヤーのリグレットR_Tを最小化

$$\mathrm{R}_T := \sum_{t=1}^{T} \boldsymbol{\ell}_t^\top \mathbf{1}_{X_t} - \min_{X \in \mathcal{S}} \sum_{t=1}^{T} \boldsymbol{\ell}_t^\top \mathbf{1}_X$$

敵対者が定める
各アームの損失ベクトル

プレイヤーが選択したX_t
のアームの組み合わせ指示ベクトル

アーム：[d] = {1,…,d}

アーム集合：E = [d]

選択肢集合：$S \subseteq 2^E$

スーパーアーム：$X_t \in S$

ラウンド： t = {1,…,T}

損失ベクトル： $l_t = (l_{t,1}, \dots, l_{t,i})$

指示ベクトル：$1_{Xt} \in \{0,1\}^d$

問題設定　リグレットとは？

リグレットR_T

$$\mathrm{R}_T := \sum_{t=1}^{T} \boldsymbol{\ell}_t^\top \mathbf{1}_{X_t} - \min_{X \in \mathcal{S}} \sum_{t=1}^{T} \boldsymbol{\ell}_t^\top \mathbf{1}_X$$

選択結果の
累積コスト

最小コストの選択肢
の総コスト

→理想の選択結果からどのくらい損しているか、損失具合を表現

アーム：[d] = {1,…,d}

アーム集合：E = [d]

選択肢集合：$S \subseteq 2^E$

スーパーアーム：$X_t \in S$

ラウンド： t = {1,…,T}

損失ベクトル： $l_t = (l_{t,1}, \dots, l_{t,i})$

指示ベクトル：$1_{Xt} \in \{0,1\}^d$

既往　Combandアルゴリズム

・リグレット上界

$Comband$アルゴリズムで達成されるリグレットは$O(\sqrt{n})$

n:最大ラウンド数

実用上、試行回数は不明な場合がある

→各ラウンドtの期待リグレット$O(\sqrt{t})$に改善:COMBD

・効率性

共起確率行列P_tの計算規模は$O(d^2|S|)$

|S|= 2^d :選択肢集合, d:アーム数

→ ZDDを用いることで計算規模削減:COMBD3

ComBDアルゴリズム Comband with Decreasing Parameters

特徴
・ $Comband$アルゴリズムと流れは同様
・ハイパーパラメータαを導入し、パラメータγ, ηを削減

アルゴリズムの流れ

①ハイパーパラメータαからγ, ηを設定

②プレイヤーが、各選択肢の選択確率p_tからスーパーアームX_tを選択

③敵対者によりコストベクトルが設定され、プレイヤーはコストを受け取る
（プレイヤーはコストベクトルの詳細を知らない）

④共起確率行列$P_t(i,j)$を計算

⑤プレイヤーはコストを元に各アームのコストの不偏推定量を算出

⑥各アームの重み$w_{t+1,i}$を更新し、①に戻る

ComBDアルゴリズム Comband with Decreasing Parameters

アルゴリズム　パラメータ(α, S)
初期設定：各アームのコスト和$\hat{L}_{0,i} = 0$, 重み$w_{1,i} = 1$

①ハイパーパラメータαからγ, ηを設定

$$\gamma_t = \frac{t^{-\frac{1}{\alpha}}}{2}, \eta_{t+1} = \frac{\lambda(t+1)^{-\frac{1}{\alpha}}}{2D^2}$$

$$D = \max_{X \in S} ||1_X||$$

λ：$E_{X\sim u}\left[1_{Xt}1_{Xt}^T\right]$の最小非ゼロ固有値
u： S上の一様分布

②プレイヤーが、各選択肢の選択確率p_tからスーパーアームX_tを選択

$$p_t(X_t) = (1-\gamma_t)p(X_t; w_t, S) + \gamma_t p(X_t; 1_E, S)$$

Sと$w_t = \left(w_{t,1}, \dots, w_{t,d}\right)^T$による制約付き分布　　S上の一様分布

$$p(X; w, S) = \frac{w(X)}{Z(w,S)}, w(X) = \prod_{i \in X} w_i,$$

$$Z(w,S) = \sum_{X \in S} w(X), w = (w_1, \dots, w_d)^T$$

ComBDアルゴリズム Comband with Decreasing Parameters

③敵対者によりコストベクトルが設定され、プレイヤーはコストを受け取る

$$c_t = l_t^T 1_{Xt}$$

④共起確率行列$P_t(i,j)$を計算
行列Pの(i,j)成分$P(i,j)$が

$$p(i \in X, j \in X) = \sum_{X \in S: i,j \in X} p(X)$$

で与えられるとき、 Pを共起確率行列という
ラウンドtにおける共起確率行列のi,j成分$P_t(i,\mathrm{j})$は②の式と合わせて

$$P_t(i,j) = (1-\gamma_t)p(i \in X, j \in X; w_t, S) + \gamma_t p(i \in X, j \in X; 1_E, S)$$

Sと$w_t = \left(w_{t,1}, \ldots, w_{t,d}\right)^T$による制約付き分布　　S上の一様分布

⑤プレイヤーはコストを元に各アームのコストの不偏推定量を算出

$$\tilde{l_t} = c_t P_t^{-1} 1_{Xt}$$

各アームのコスト和の更新

$$\hat{L}_{t,i} = \hat{L}_{t-1,i} + \tilde{l}_{t,i}$$

⑥各アームの重み$w_{t+1,i}$を更新し、①に戻る

$$w_{t+1,i} = \exp\left(-\eta_{t+1}\hat{L}_{t,i}\right)$$

ComBDアルゴリズム Comband with Decreasing Parameters

$Theorem\ 1$

$COMBD(\alpha = 3, S)$ のとき

$$R_T \leq O\left(\left(\frac{d\lambda}{L^2} + \sqrt{\frac{L^2}{\lambda}\ln\frac{|S|+2}{\delta}}\right)T^{\frac{2}{3}}\right)$$

を少なくとも、確率$1-\delta$で達成する

$Theorem\ 2$

$COMBD(\alpha = 2, S)$のとき、

$$\bar{R}_T \leq O\left(\left(\frac{d\lambda}{L^2} + \frac{L^2 \ln|S|}{\lambda}\right)\sqrt{T}\right)$$

を達成する

ComBD3アルゴリズム

$COMBD$アルゴリズムは$R_T \leq O(\sqrt{T})$を達成しているが、

②の $p_t(X_t)$の計算時のサンプリング

④共起確率行列P_tの計算

は計算規模が$O(d|S|), O(d^2|S|)$　Sは2^d通りの組み合わせ数

→ZDDを用いた動的計画法によって効率化した$COMBD3$を提案

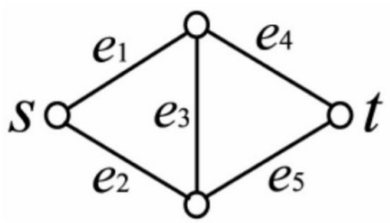
　計算規模を$O(|V|), O(d|V|)$に改善

ComBD3アルゴリズム　ZDD

数え上げ方

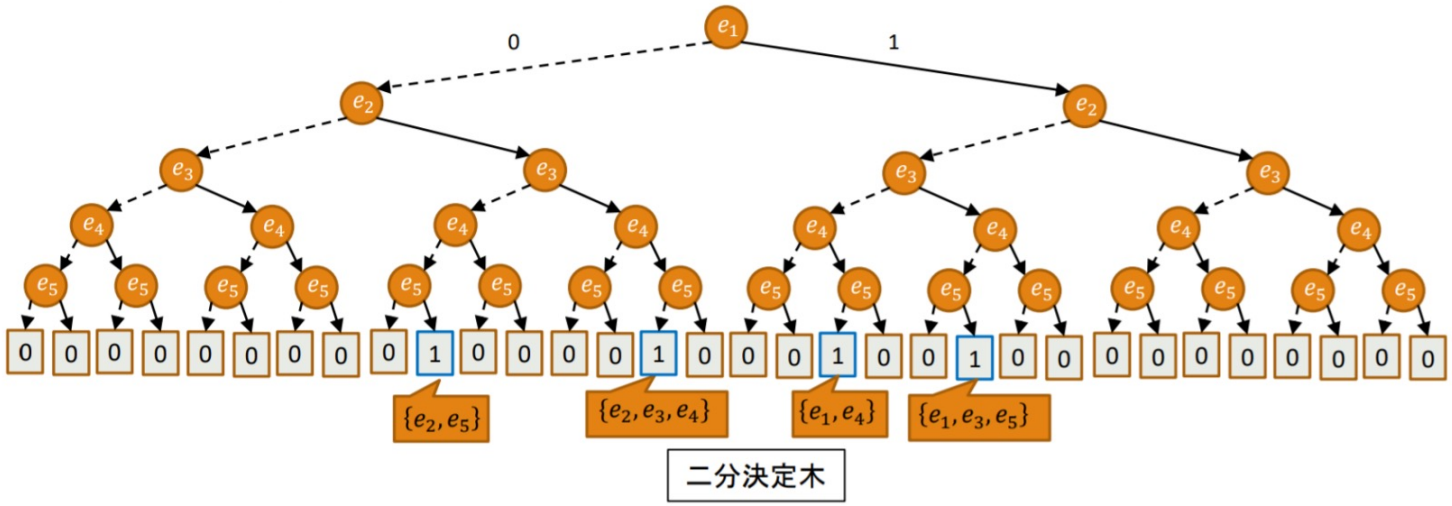
各リンクを経路として選択するかどうかの2通りの決定の連続として
場合分け二分木で考える

例)

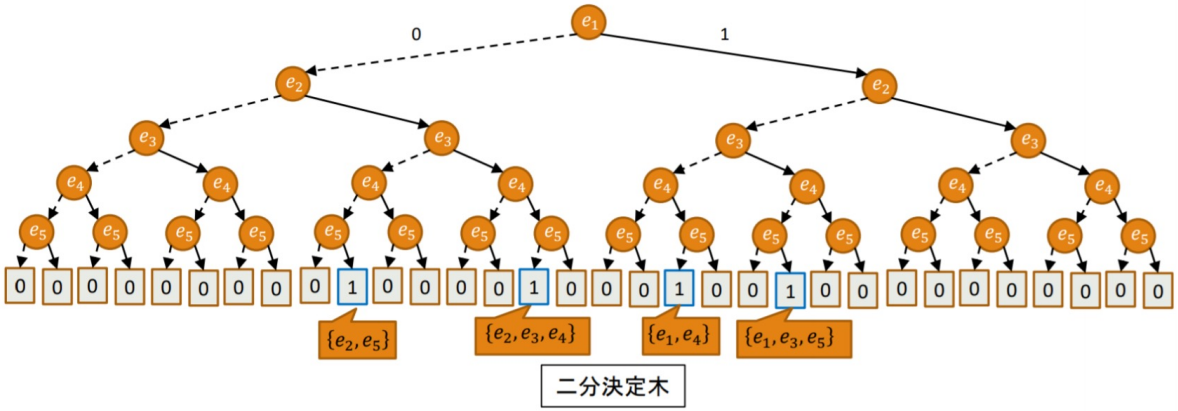


$s \to t$の可能経路集合

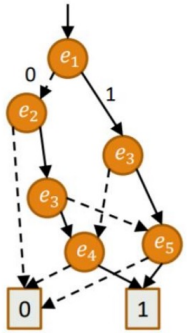
$\{e_1, e_4\}, \{e_1, e_3, e_5\}$
$\{e_2, e_5\}, \{e_2, e_3, e_4\}$



ComBD3アルゴリズム　ZDD



11 × 11の格子グラフの
対角2頂点を連結する経路列挙問題
では290億年から数秒に短縮が可能



省略や節点の共有により
ネットワークを圧縮した表現方法

ComBD3アルゴリズム　ZDD

$G_s = (V, A)$
V = {0,1, … |V| − 1}：頂点集合
A ⊆ V×V：枝集合

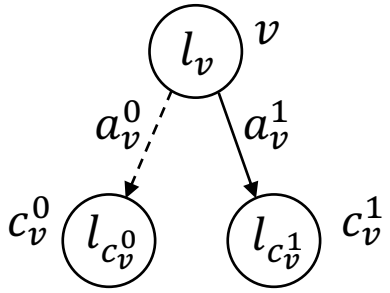
$$X(R) := \{l_v \in E \mid (v, \mathrm{c}_v^1) \in R\}$$

→頂点vからb-終端=1のあるパスのラベル集合

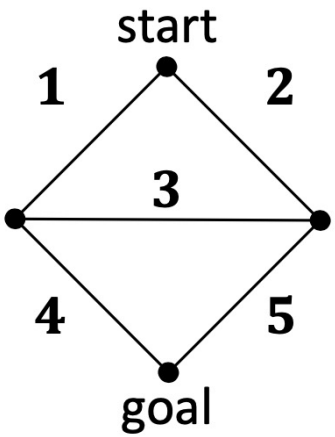
$$\mathcal{S} = \{X(R) \mid R \in \mathcal{R}_{r,1}\}.$$

→頂点rからb-終端=1の全てのパスのラベル集合

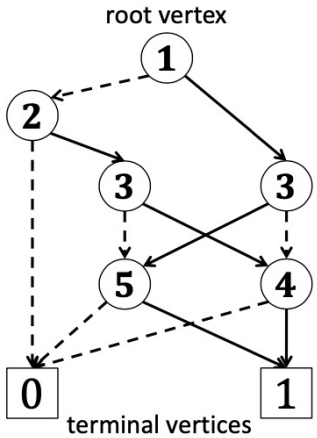
$\mathcal{R}_{v,u}$ ：頂点vからuへのG_s上のパス集合
l_v：頂点vのラベル
a_v^b：頂点vから伸びる枝（b-枝）
c_v^b：頂点vのb-枝の先にある頂点（b-子）
$X(R)$：パスR中に含まれる頂点ラベル集合



文字定義



例) ノードエッジグラフ



ZDD表現

$$\mathcal{S} = \{X(R) \mid R \in \mathcal{R}_{r,1}\}$$
= {{1,4},{2,5},{1,3,5},{2,3,4}}

ComBD3アルゴリズム

説明の流れ

・制約付き分布$p(X;w,S)$からのサンプリング
①Forward weightの計算
②Backward weightの計算
③制約付き分布$p(X;w,S)$からのXのサンプリングアルゴリズム

・共起確率行列計算
④共起確率 $\mathrm{P}_{i,i}$ の計算
⑤共起確率 $\mathrm{P}_{i,j}$ の計算
⑥Backward weightに条件を加えた$C_{v,j}$の計算
⑦共起確率行列$P(i,j)$の計算

ComBD3アルゴリズム

制約付き分布からのサンプリング

前向き、後ろ向きアルゴリズムにより各パスの重みの計算

Forward weight(前向き)

$$\mathrm{F}_v := \sum_{R \in \mathcal{R}_{r,v}} w(R)$$

頂点rからvへのパスの重み和

Backward weight(後ろ向き)

$$\mathrm{B}_v := \sum_{R \in \mathcal{R}_{v,1}} w(R)$$

頂点vから1へのパスの重み和

$w(R) = w(X(R)) = \prod_{i \in X(R)} w_i$：あるパス$R$内の各ラベル重み$w_i$の積

ZDD上の動的計画法を利用し、
vから1までの$F = \{F_0, .., F_r\}, B = \{B_0, \ldots, B_r\}$を計算

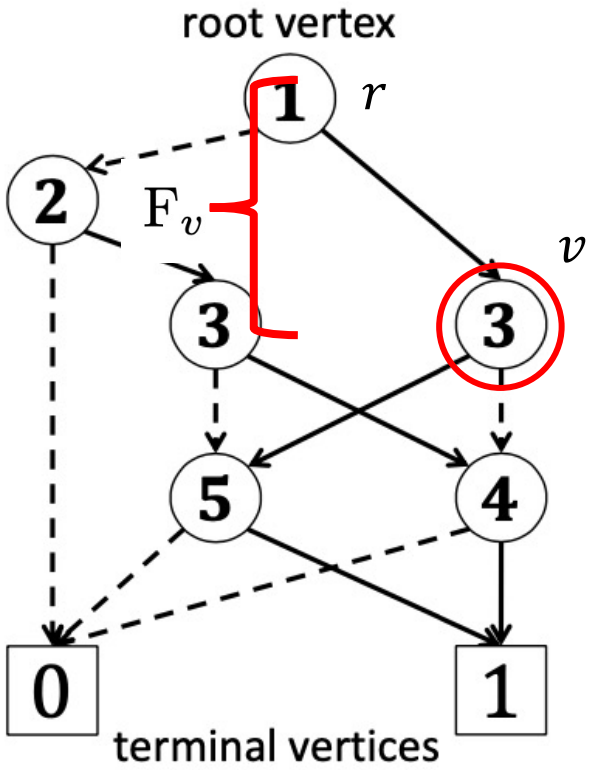
ComBD3アルゴリズム　制約付き分布からのサンプリング

前向き、後ろ向きアルゴリズムにより各パスの重みの計算

Forward weight(前向き)

$$\mathrm{F}_v := \sum_{R \in \mathcal{R}_{r,v}} w(R)$$

頂点rからvへのパスの重み和



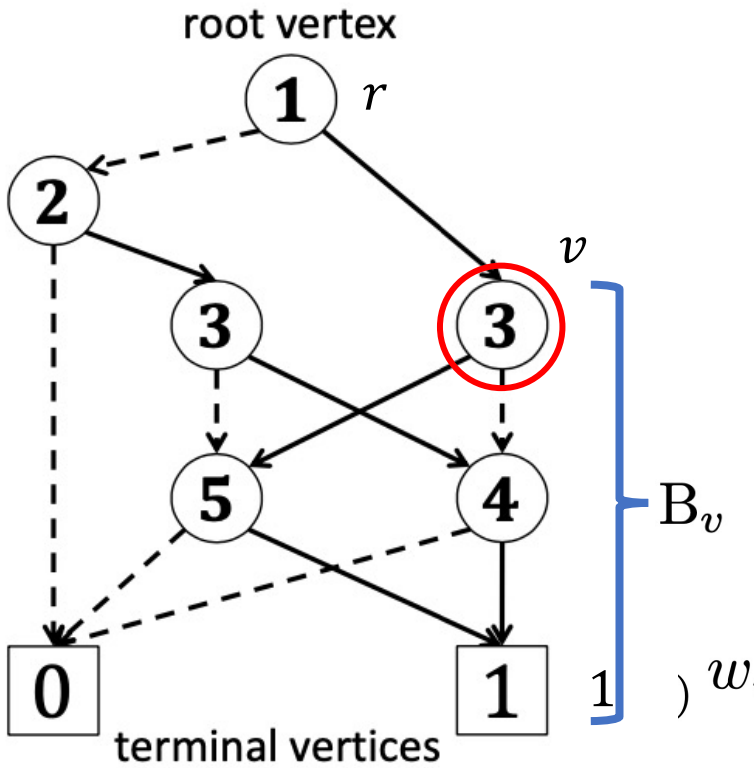
$w(R) = w(X(R)) = \prod_{i \in X(R)} w_i$:

ZDD上の動的計画法を利用し、
vから1までの$F = \{F_0, .., F_r\}, B = \{B_0, \ldots, B_r\}$ を計算

ComBD3アルゴリズム

制約付き分布からのサンプリング

前向き、後ろ向きアルゴリズムにより各パスの重みの計算



Backward weight(後ろ向き)

$$\mathrm{B}_v := \sum_{R \in \mathcal{R}_{v,1}} w(R)$$

頂点vから1へのパスの重み和

w_i：あるパスR内の各ラベル重みw_iの積

ZDD上の動的計画法を利用し、

vから1までの$F = \{F_0, .., F_r\}, B = \{B_0, \dots, B_r\}$ を計算

ComBD3アルゴリズム　制約付き分布からのサンプリング

Forward weight(G_s, w)

各ラベルl_vに与えられる重み

Step1. 初期設定　$F_r = 1, F_v = 0$

$v = r$～2まで実行

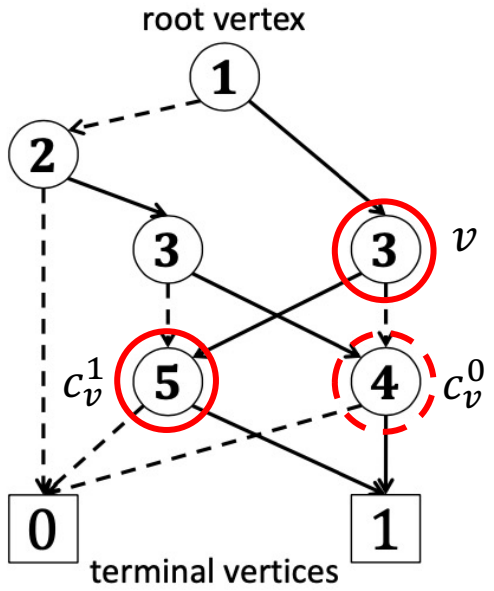
Step2.
頂点vの0-枝にある頂点c_v^0の重み更新

$$F_{c_v^0} \mathrel{+}= F_v$$

頂点vの1-枝にある頂点c_v^1の重み更新

$$F_{c_v^1} \mathrel{+}= w_{l_v} F_v$$

Return $F = \{F_0, .., F_r\}$



ComBD3アルゴリズム

制約付き分布からのサンプリング

Backward weight(G_s, w)

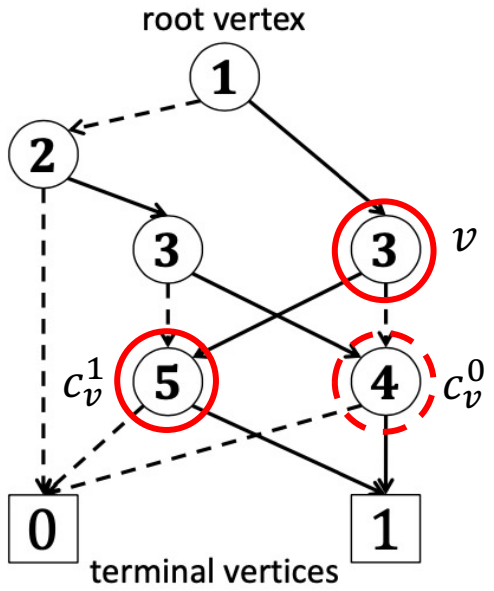
Step1. 初期設定　$B_1 = 1, B_v = 0$

$v = 2$〜rまで実行

Step2.
頂点vの0-枝先の頂点c_v^0の重みと
1-枝先の頂点の重みの和で更新

$$B_v = B_{c_v^0} + w_{l_v} B_{c_v^1}$$

Return $B = \{B_0, .., B_r\}$



ComBD3アルゴリズム　制約付き分布からのサンプリング

制約付き分布：$p(X;w,S)=\frac{w(X)}{Z(w,S)}$

パスXの重み和 / 全てパスの重み和

$Draw(G_s,w,B)$

Step1. 初期設定
選択肢$X \leftarrow \{\}$　頂点$v \leftarrow r$

Step2. ベルヌーイ分布に従い、 枝の値を算出

$$\theta \leftarrow w_{l_v}\frac{B_{c_v^1}}{B_v}, b \sim Ber(\theta)$$

ラベルl_vを通るパスの重み和 / $v \rightarrow 1$の全てパスの重み和

Step3. $b=1$ならl_vをXに追加

Step4. vの更新
$v \leftarrow c_v^b, v>1$まで$step2\sim4$を繰り返す

ComBD3アルゴリズム　共起確率行列の計算

制約付き分布$p(X;\boldsymbol{w},\mathcal{S})$が与えられた元で、

共起確率：$P_{i,j} = p(i \in X, j \in X; w, S)$と定義

$$p(X;\boldsymbol{w},\mathcal{S}) := \frac{w(X)}{Z(\boldsymbol{w},\mathcal{S})} \text{ より}$$

$$P_{i,j} = \sum_{R \in R_{r,1}: i,j \in X(R)} \frac{w(R)}{Z(w,S)}$$

この計算式を愚直に解くと$O(d^2|V|)$の計算規模

→Forward WeightとBackward Weightを用いることで、$O(d|V|)$に削減

ComBD3アルゴリズム　共起確率行列の計算

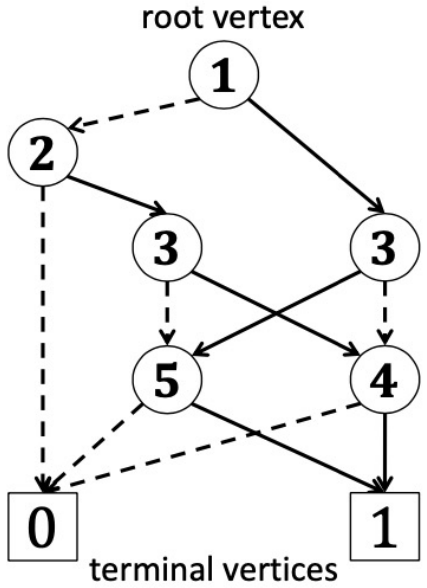
まず、$P_{i,i}$を考える

$$\mathrm{P}_{i,i} = \sum_{R \in \mathcal{R}_{r,1} : i \in X(R)} \frac{w(R)}{Z(\boldsymbol{w}, \mathcal{S})}$$

Rをforwardとbackwardに分解

$R: r \sim 1 \rightarrow R': r \sim v + \{i\} + R'': c_v^1 \sim 1$

$$= \sum_{v \in V : l_v = i} \sum_{\substack{R' \in \mathcal{R}_{r,v} \\ R'' \in \mathcal{R}_{\mathrm{c}_v^1,1}}} \frac{w(R' \cup \{i\} \cup R'')}{\mathrm{B}_r} \quad \because Z(\boldsymbol{w}, \mathcal{S}) = \mathrm{B}_r$$



Forward weight F_vとBackward weight B_vに置き換え

$$= \sum_{v \in V : l_v = i} \frac{\mathrm{F}_v w_i \mathrm{B}_{\mathrm{c}_v^1}}{\mathrm{B}_r} \quad \because \mathrm{F}_v = \sum_{R \in \mathcal{R}_{r,v}} w(R) \text{ and } \mathrm{B}_v = \sum_{R \in \mathcal{R}_{v,1}} w(R).$$

ComBD3アルゴリズム　共起確率行列の計算

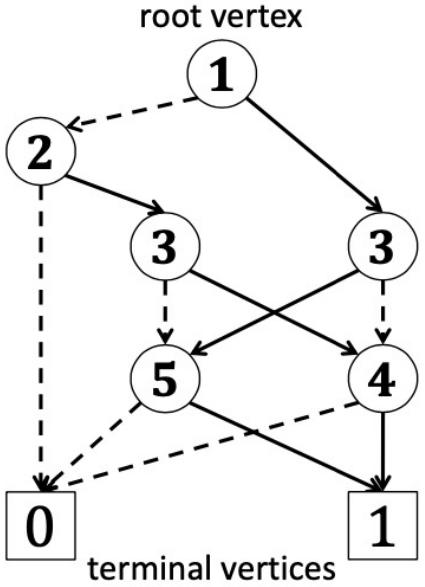
次に、$P_{i,j}$を考える

$$\mathrm{P}_{i,j} = \sum_{R \in \mathcal{R}_{r,1} : i,j \in X(R)} \frac{w(R)}{Z(\boldsymbol{w}, \mathcal{S})}$$

Rをforwardとbackwardに分解

$R: r \sim 1 \rightarrow R': r \sim v + \{i\} + R'': c_v^1 \sim 1$

$$= \sum_{v \in V : l_v = i} \sum_{\substack{R' \in \mathcal{R}_{r,v} \\ R'' \in \mathcal{R}_{c_v^1,1} : j \in X(R'')}} \frac{w(R' \cup \{i\} \cup R'')}{\mathrm{B}_r} \quad \because Z(\boldsymbol{w}, \mathcal{S}) = \mathrm{B}_r$$



Forward weight F_vと$C_{v,j}$に置き換え

$$= \sum_{v \in V : l_v = i} \frac{\mathrm{F}_v w_i \mathrm{C}_{c_v^1, j}}{\mathrm{B}_r} \quad \because \mathrm{F}_v = \sum_{R \in \mathcal{R}_{r,v}} w(R) \text{ and } \mathrm{C}_{v,j} = \sum_{R \in \mathcal{R}_{v,1} : j \in X(R)} w(R).$$

$C_{v,j}$: B_vに「ラベルjを通る」という条件が加わった重み和

→BWCというアルゴリズムで求める

ComBD3アルゴリズム　共起確率行列の計算

$C_{v,j}$: B_vに「ラベルjを通る」という条件が加わった重み和

1: **Algorithm** BWC($\mathbf{G}_S, \boldsymbol{w}, \mathrm{B}$)
2: $\mathrm{C}_{v,j} \leftarrow 0\ (\forall v \in V, \forall j \in E)$
3: **for** $v = 2, \ldots, r$ **do**
4: $\quad \mathrm{C}_{v,l_v} \leftarrow w_{l_v} \mathrm{B}_{\mathrm{c}_v^1}$
5: $\quad$ **for** $j = l_v + 1, \ldots, d$ **do**
6: $\quad\quad \mathrm{C}_{v,j} \leftarrow \mathrm{C}_{\mathrm{c}_v^0,j} + w_{l_v} \mathrm{C}_{\mathrm{c}_v^1,j}$
7: **return** $\mathrm{C} := \{\mathrm{C}_{v,j} \mid v \in V, j \geq l_v\}$

ComBD3アルゴリズム　共起確率行列の計算

$C_{v,j}$: B_vに「ラベルjを通る」という

1: **Algorithm** BWC(**G**

2: $\mathrm{C}_{v,j} \leftarrow 0$ $(\forall v \in V, \forall j$

3: **for** $v = 2, \ldots, r$ **do**

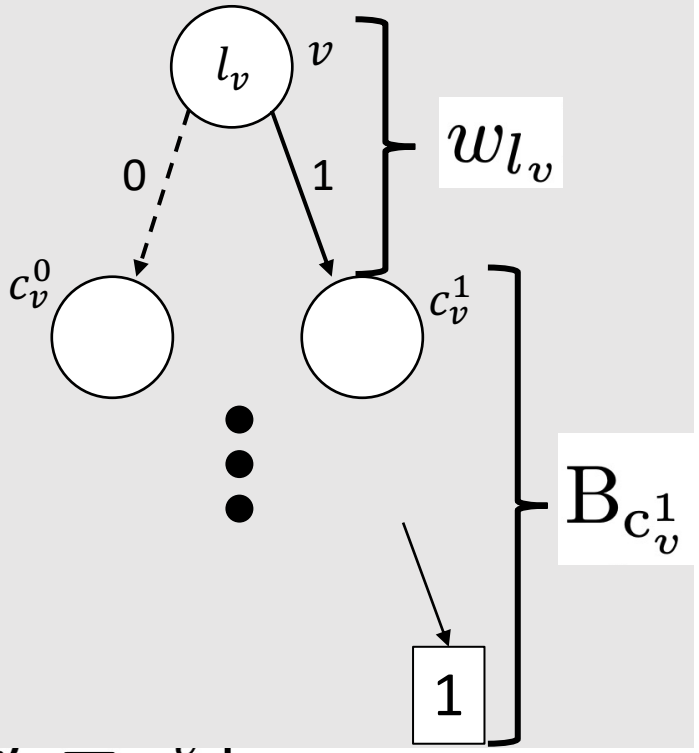
4: $\mathrm{C}_{v,l_v} \leftarrow w_{l_v} \mathrm{B}_{\mathrm{c}_v^1}$

5: **for** $j = l_v + 1, \ldots$

6: $\mathrm{C}_{v,j} \leftarrow \mathrm{C}_{\mathrm{c}_v^0,j}$ −

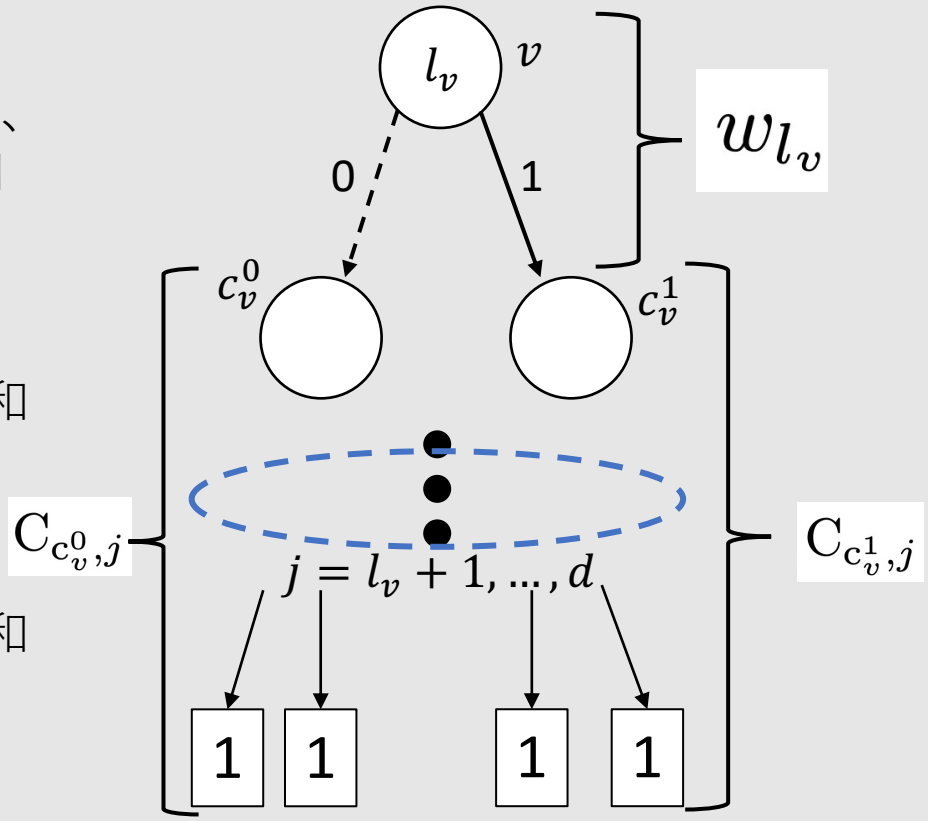
7: **return** $\mathrm{C} := \{\mathrm{C}_{v,j} \mid$

C_{v,l_v}は、 $v \to 1$のパスのうち、
ラベルl_vを通るパスの重み和



ComBD3

$C_{v,j}$は、 $v \to 1$のパスのうち、
ラベルjを通るパスの重み和
↑
vから$b-$枝$=0$の
頂点$c_v^0 \to 1$のパスのうち、
ラベルjを通るパスの重み和
＋
vから$b-$枝$=1$の
頂点$c_v^1 \to 1$のパスのうち、
ラベルjを通るパスの重み和



$C_{v,j}$: B_vに

Alg

$\mathrm{C}_{v,j}$

for

for $j = l_v + 1, \ldots, d$ **do**

$\mathrm{C}_{v,j} \leftarrow \mathrm{C}_{c_v^0,j} + w_{l_v} \mathrm{C}_{c_v^1,j}$

return $\mathrm{C} := \{\mathrm{C}_{v,j} \mid v \in V, j \geq l_v\}$

ComBD3アルゴリズム　共起確率行列の計算

$$\mathrm{P}_{i,j} = \sum_{R \in \mathcal{R}_{r,1}: i,j \in X(R)} \frac{w(R)}{Z(\boldsymbol{w}, \mathcal{S})}$$

$$= \sum_{v \in V: l_v = i} \frac{\mathrm{F}_v w_i \mathrm{C}_{\mathrm{c}_v^1, j}}{\mathrm{B}_r} \qquad \because \mathrm{F}_v = \sum_{R \in \mathcal{R}_{r,v}} w(R) \text{ and } \mathrm{C}_{v,j} = \sum_{R \in \mathcal{R}_{v,1}: j \in X(R)} w(R).$$

→共起確率行列の計算アルゴリズムCPM

ComBD3アルゴリズム　共起確率行列の計算

Algorithm $\mathrm{CPM}(\mathbf{G}_{\mathcal{S}}, \boldsymbol{w}, \mathrm{B}, \mathrm{F}, \mathrm{C})$
$\mathrm{P}_{i,j} \leftarrow 0\ (\forall i, j \in E)$
for $v = 2, \ldots, r$ **do**
    $i \leftarrow l_v$
    $\mathrm{P}_{i,i} += \mathrm{F}_v w_i \mathrm{B}_{\mathrm{c}_v^1} / \mathrm{B}_r$
    for $j = i+1, \ldots, d$ **do**
        $\mathrm{P}_{i,j} += \mathrm{F}_v w_i \mathrm{C}_{\mathrm{c}_v^1, j} / \mathrm{B}_r$
return $\mathrm{P} := \{\mathrm{P}_{i,j} \mid i, j \in [d], i \leq j\}$

計算規模は、$O(d|V|)$に改善

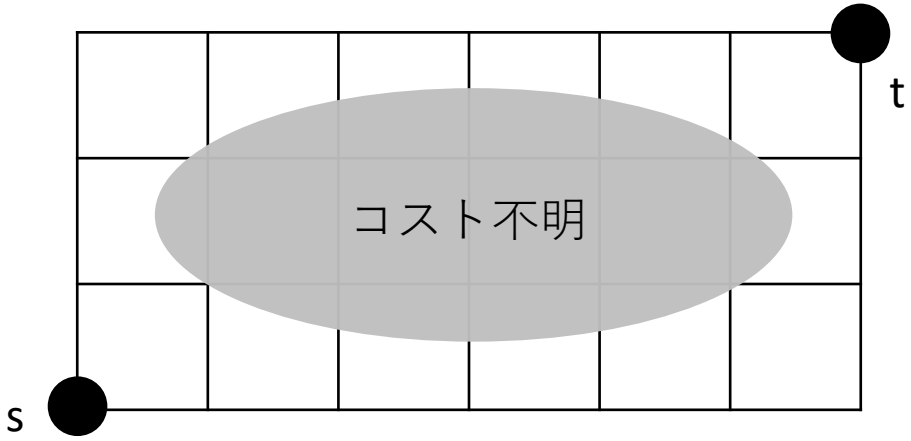
数値実験　OSP

オンライン最短経路問題(OSP)

・NWの状況が不明な中で最短経路を探索する問題

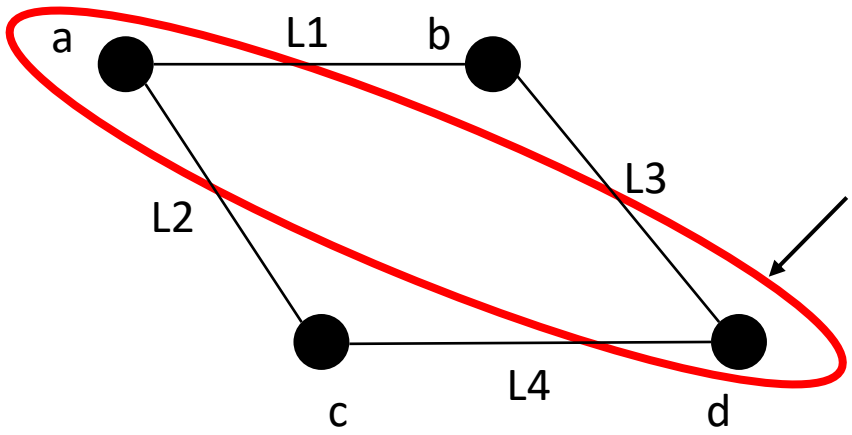
・3×m（m=3~10）のグリッドNWで計算

・対角線上のs,tを始点、終点とする



数値実験　DST

・動的シュタイナー木問題DST
部分集合のノードを含む木をシュタイナー木といい、
その木のエッジにかけられる重み（コスト）の最小化を目指す問題



頂点集合：V={a,b,c,d}
部分集合： T={a,d}
シュタイナー木： S={{L1,L3},{L2,L4}}

・3×m（m=3~10）のグリッドNWで計算
・四隅のノードを部分集合とするシュタイナー木として、最小化問題をとく

数値実験　損失ベクトルの設定

損失ベクトルの設定

まず、ベクトル$\boldsymbol{\mu_0}$を$[0,1]^d$上の一様分布からサンプリング(dはグリッド数)

次に、各ラウンドtで　確率0.9で$\boldsymbol{\mu_t = \mu_{t-1}}$
確率0.1で$\boldsymbol{\mu_t}$を同様の一様分布からサンプリング

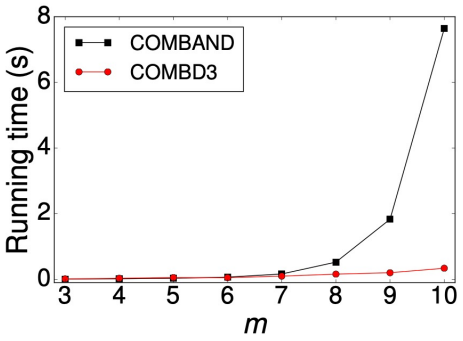
各iに対して、 $h_i \sim Ber(\mu_{t,i})$をひき　$h_i = 1$なら$l_{t,i} = \frac{1}{d}$
$h_i \neq 1$なら$l_{t,i} = -\frac{1}{d}$

OSP,DSTを提案したCOMBD3とCOMBANDで計算速度とリグレットを比較

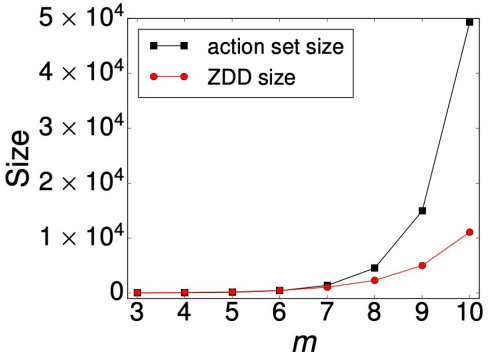
数値実験　ComBD3 VS Comband

計算速度について
・グリッド数が増えるに従い、
計算速度に大きな差が発生

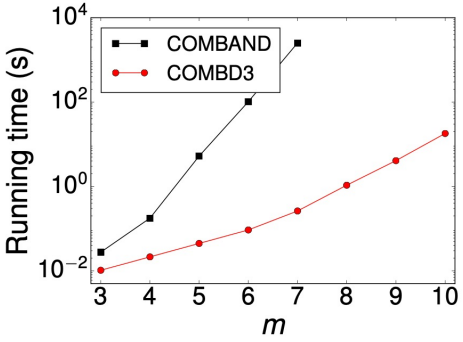
・ZDDによって、NWサイズを削減
DSTでは、m=10で10^5のオーダーの差が発生



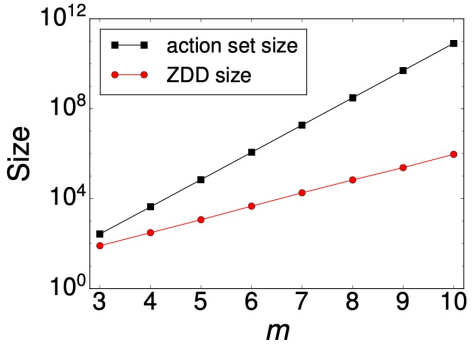
(a) OSP



(c) OSP



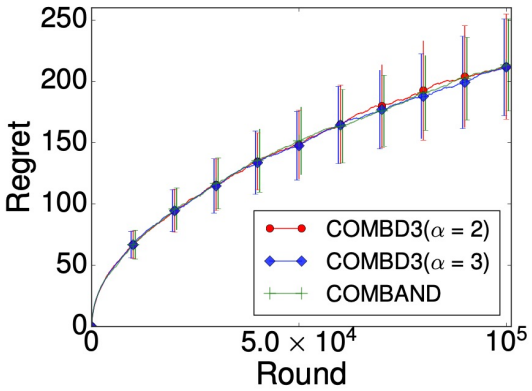
(b) DST (semi-log)



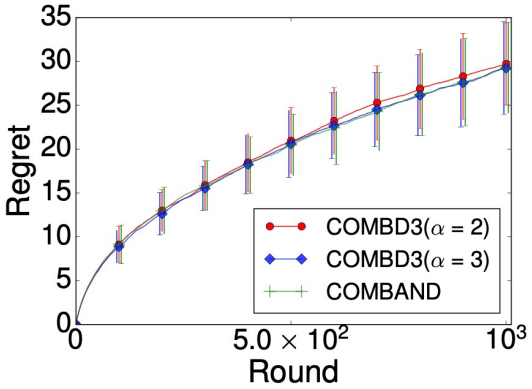
(d) DST (semi-log)

数値実験　ComBD3 VS Comband

リグレットについて
・提案手法のリグレットがcombandと
　同等のリグレット
・どの手法も劣線形のリグレット



(e) OSP



(f) DST

数値実験　CG

混合ゲーム（Congestion Game）
複数人のプレイヤーがいた場合のOSPのこと

m人のプレイヤーと始点sと終点tのNWの元で
プレイヤーたちは、s→tに同時メッセージを送信する

T回繰り返し、各プレイヤーはT回の累積送信時間の最小化を行う

各辺は、同じ辺を使えば使うほど、通過に時間がかかる
→お互いが敵対者という認識

数値実験　CG　問題設定

$X_t^k \in S(k \in [m])$： k番目のプレイヤーがラウンドtに選んだパス
$X_{t,i}^k \in \{0,1\}$を$1_{X_t^k}$のi番目の成分
$l_{t,i}^k$： k番目のプレイヤーがラウンドtに辺iを使った際の損失

$$l_{t,i}^k = \beta_i \kappa^{N_{t,i}^{-k}}$$

辺の長さ

遅延を表す定数

$$N_{t,i}^{-k} = \sum_{k' \neq k} X_{t,i}^{k'}$$
同時にiを使った敵数

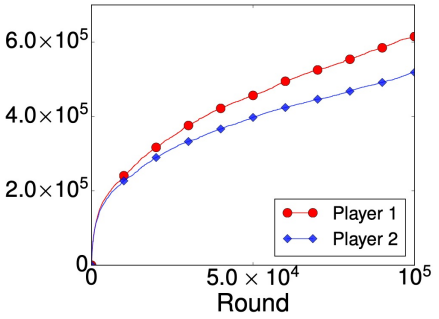
$m = 2, \kappa = 10$とした
NWには、Internet topology zoo(http://www.topology-zoo.org/)の
MCIネットワークとATTネットワークを用いた

	# nodes	# edges	# s-r paths	ZDD size
MCI	19	33	1,444	756
ATT	25	56	213,971	37,776

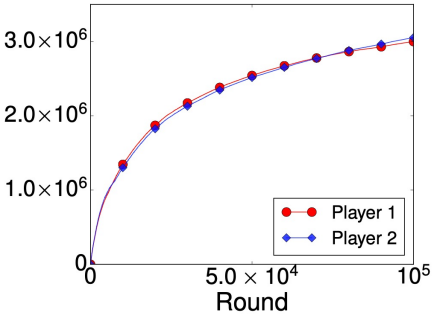
NW情報

数値実験　CG　結果

・各NWに対するリグレット
各プレイヤーは、劣線形なリグレット



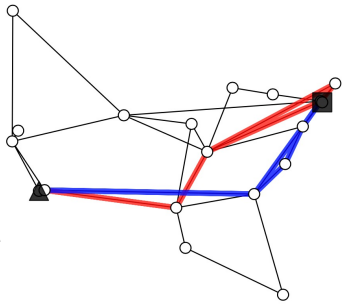
(g) CG on MCI



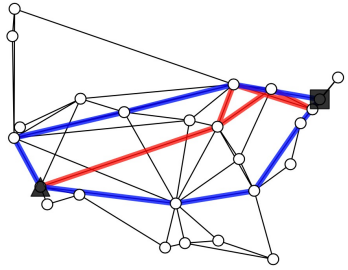
(h) CG on ATT

・各プレイヤーの頻出パス
それぞれの頻出パスが分かれるように
選択されている。
→混雑(コスト)を避けることの確認

・バンディット型のルーティング問題に対応
選択した選択肢のコストのみがわかる状況
でも、最適な状況になることを確認



(a) MCI



(b) ATT

結論

Combandアルゴリズムの改善

・ZDDを用いることで、選択肢集合をコンパクト化させ、$O(d^2|V|)$から$O(d|V|)$に計算規模を削減

・共起確率行列、制約付き分布からのサンプリングを動的計画法に基づく方法によって計算し、$O(\sqrt{T})$または$O(\sqrt{T^{2/3}})$のリグレットを達成

実用可能な計算効率

・数値実験によって、劣線形のリグレットを達成し、ルーティング問題にも対応できることを示した。

参考文献

N. Cesa-Bianchi and G. Lugosi.
Combinatorial bandits. J. Comput. Syst. Sci., 78(5):1404 – 1422, 2012.

(参考): 本多淳也, & 中村篤祥. (2016)『バンディット問題の理論とアルゴリズム』