The 28th Annual Conference of the Japanese Society for Artificial Intelligence, 2014

1D4-OS-11a-5

乗算型重み更新法に基づく分散制約最適化アルゴリズム

Distributed Multiplicative Weights Methods for DCOP

波多野大督 *1*2　吉田悠一 *1
Daisuke HATANO　Yuichi YOSHIDA

*1国立情報学研究所
National Institute of Informatics

*2JST, ERATO, 河原林巨大グラフプロジェクト, ビッグデータ数理国際研究センター
JST, ERATO, Kawarabayashi Large Graph Project, Global Research Center for Big Data Mathematics

We introduce a new framework for solving distributed constraint optimization problems (DCOP) that extends the domain of each variable into a simplex. We propose two methods for searching the extended domain for good assignments. The first one relaxes the problem using a linear programming, finds the optimum LP solution, and then round it to an assignment. The second one plays a cost-minimization game, finds a certain kind of equilibrium, and then round it to an assignment. Both methods are realized by performing the multiplicative weights method in a distributed manner. We experimentally demonstrate that our methods have a good scalability, and especially the second method outperforms existing algorithms in terms of the solution quality and the efficiency.

はじめに

近年，AI コミュニティで computational sustainability プロジェクトが発足された．それにより，スマートグリッド等の巨大な問題を分散かつ強調的に扱う必要性が急速に拡大している．**分散制約最適化問題** (distributed constraint optimization problem, 以下 DCOP とする) は分散協調問題解決において盛んに研究されている．その目的は，全体のコストを最小とする値の割当を探索することである．

本論文では，DCOP の非厳密解法に焦点を置き，新たなアプローチを提案する．提案アプローチでは，変数の値域を d 次元のシンプレックスに拡張する．ここで，d は値域のサイズとする．拡張した問題に対し良い割当を探索する解法を，ここで提案する．提案解法は，二つあり，それぞれ乗算型重み更新法（multiplicative weights method，以下 MW 法とする）に基づく．MW 法は機械学習，最適化，ゲーム理論等に広く応用されている [Arora 12].

一つ目は線形計画問題（linear programming，以下 LP とする）を利用する解法で，DMW-LP (a distributed multiplicative weights method for solving linear programmings) と呼ぶ．この解法では，エージェントは線形緩和された DCOP 問題例を協調的に解く．その後，得られた LP の解を整数解へと丸める．LP は凸最適化問題であるため，MW 法により最適解に収束するが，分散環境で動作するように解法を修正する必要がある．また，最小カット問題等のある種のクラスの COP に対しては LP の解から最適解を復元できる．

二つ目の解法は，**コスト最小化ゲーム**として DCOP を解くことから DMW-Game (a distributed multiplicative weights method for solving games) と呼ぶ．このゲームでは，各プレイヤーは変数に対応しており，値域の各値に対し確率分布を割り当てる．その目的は *regret* の最小化である．regret とは，最良の単一戦略から得られるコストと確率分布から得られる平均期待コストの差である．MW 法を用いることで各エージェントの regret を任意の小さい値にできる．最後に，得られた確率分布を整数値に丸める．

連絡先: 波多野大督，hatano@nii.ac.jp,
国立情報学研究所 ビッグデータ数理国際研究センター，
JST, ERATO, 河原林巨大グラフプロジェクト，
〒 101-8430　東京都千代田区一ツ橋 2-1-2

提案解法に対し，最先端の非厳密解法である MaxSum と DeQED を用いて評価実験した．実験結果より，提案解法は他の解法と較べて問題サイズに対するスケーラビリティがあることが示された．特に，DMW-Game は他の解法と比較して効率的に良い質の解が得られた．

準備

正の整数 t に対して，$[t]$ は集合 $\{1, 2, \ldots, t\}$ を表す．また，太文字はベクトルとして使用する．二つのベクトル $\boldsymbol{x}$ と $\boldsymbol{y}$ に対して，$\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ は内積を表す．値域 D 上のある確率分布 $\mathcal{D}$ に対して，$a \sim \mathcal{D}$ は分布 $\mathcal{D}$ から値 $a \in D$ をサンプリングすることを意味する．

分散制約最適化問題

制約最適化問題（constraint optimization problem，以下 COP とする）は，**変数集合** $X = \{x_1, x_2, \ldots, x_n\}$ と変数上の枝集合 E，**コスト関数集合** $F = \{f_{i,j}\}_{(i,j)\in E}$ で定義される．ここで，変数 $x_i \in X$ は有限の**値域** D_i をもち，D_i から値を選択する．また，コスト関数 $f_{i,j} : D_i \times D_j \to \mathbb{R}^+$ は非負のコストを返す．COP の目的は**コストの総和** $f(\boldsymbol{x}) := \sum_{(i,j)\in E} f_{i,j}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ が最小となる変数への値の割当 $\boldsymbol{x} \in D := D_1 \times D_2 \times \cdots \times D_n$ を探索することである．エージェント $i \in [n]$ に対して $d_i = |D_i|$ とし，すべての値域のうち最大のものを $d_{\max} = \max_{i\in[n]} d_i$, $d = |D| = d_1 d_2 \cdots d_n$ と定義する．

各エージェント i に対して，$f_i : D \to \mathbb{R}^+$ を $f_i(\boldsymbol{x}) = \sum_{j\in[n]:(i,j)\in E} f_{i,j}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ と定義する．この関数は x_i に関わるコスト関数の総和を表す．ここで，$f(\boldsymbol{x}) = \frac{1}{2}\sum_i f_i(\boldsymbol{x})$ であることに注意されたい．本論文では，すべての $i \in [n]$ と $\boldsymbol{x} \in D$ に対し，$f_i(\boldsymbol{x}) \in [0, 1]$ と仮定する．これに該当しない場合は，その変数が関わるコスト関数の数と最大のコストをかけたもので割ることにより正規化できる．

DCOP は各変数に対してその値を管理するエージェントが付

Algorithm 1 乗算型重み更新法

Fix $\eta \leq 1/2$ and set $\boldsymbol{w}_i^1 \leftarrow (1, 1, \ldots, 1)$.
for $t = 1, 2, ..., T$ **do**
  Choose decision a with probability $\boldsymbol{x}_a^t := \frac{\boldsymbol{w}_a^t}{\|\boldsymbol{w}^t\|_1}$.
  Observe the costs of the decisions $\boldsymbol{c}_1^t, \ldots, \boldsymbol{c}_d^t$.
  For every decision a, set $\boldsymbol{w}_a^{t+1} \leftarrow \boldsymbol{w}_a^t(1 - \eta \boldsymbol{c}_a^t)$.

与された COP である．エージェントは枝を共有する他のエージェントと情報を交換できる．つまり，$i \in [n]$ に対して，x_i に隣接する変数の集合を $N(i) = \{j \in [n] : (i, j) \in E\}$ と表現する．あるラウンドにおいて，エージェント i は $N(i)$ のエージェントに対して情報の送受信を行える．DCOP の目的はコストの総和が最小となる割当を探索することである．

乗算型重み更新法

以下の状態を考える．d 個の意思決定からなる集合 D が存在し，各ラウンドにおいて，意思決定を一つ選択する．具体的には，ラウンド t において，$\sum_{a \in D} \boldsymbol{x}_a^t = 1$ を満たすベクトル $\boldsymbol{x}^t = (\boldsymbol{x}_1^t, \ldots, \boldsymbol{x}_d^t)$ を選択する．$\mathcal{D}^t$ を $\boldsymbol{x}^t$ に関わる確率分布とすると，意思決定 $a \in D$ の確率分布は $\mathcal{D}^t(a) = \boldsymbol{x}_a^t$ と書ける．この確率分布 $\mathcal{D}^t$ から意思決定 a をサンプリングする．意思決定をサンプリング後，すべての意思決定に対するコストがベクトル $\boldsymbol{c}^t = (\boldsymbol{c}_1^t, \ldots, \boldsymbol{c}_d^t)$ の形で表れる．そのため，この解法から得られる期待値は $\mathbf{E}_{a \sim \mathcal{D}^t}[\boldsymbol{c}_a^t] = \langle \boldsymbol{x}^t, \boldsymbol{c}^t \rangle$ と表せる．つまり，T ラウンド後の期待値の総和は $\sum_{i=1}^T \langle \boldsymbol{x}^t, \boldsymbol{c}^t \rangle$ となる．

この期待値の総和が，結果的に最適な意思決定のコストと比較してそれほど悪くならない．つまり，$\min_{a \in D} \sum_{t=1}^T \boldsymbol{c}_a^t$ を達成するアルゴリズムが望まれる．アルゴリズム 1 は乗算型重み更新法（MW 法）と呼ばれ，この性質を満たすものとして知られている．具体的には，以下のとおりである．

Theorem 1 ([Arora 12])**.** すべてのコストが $\boldsymbol{c}_a^t \in [-1, 1]$ であると仮定する．$T = O(\frac{\log |D|}{\epsilon^2})$，$\eta = \sqrt{\frac{\log d}{T}}$ とすると，*MW* 法は以下を保障する．T ラウンド後，任意の意思決定 a に対して，

$$\frac{1}{T}\left(\sum_{t=1}^{T} \langle \boldsymbol{c}^t, \boldsymbol{x}^t \rangle - \sum_{t=1}^{T} \boldsymbol{c}_a^t\right) \leq \epsilon.$$

を満たす.

左辺は regret と呼ばれる．もし $T \to \infty$ のときの regret が高々0 であるなら，その解法は *no-regret* **法**と呼ばれる．MW 法は no-regret 法の一例である．

DMW-LP: LP に基づく解法

LP に基づく解法である DMW-LP について説明する．

LP 定式化

COP に対する LP 定式化を示す．まず，変数 $x_i \in X$ に対して，制約 $\sum_{a \in D_i} \boldsymbol{x}_{i,a} = 1$ をもつ LP の変数 $\boldsymbol{x}_{i,1}, \ldots, \boldsymbol{x}_{i,d_i}$ を導入する．ここで，$\boldsymbol{x}_{i,a}$ は変数 x_i が値 a をとる確率を表す．つまり，変数集合 $\boldsymbol{x}_i := \{\boldsymbol{x}_{i,a}\}_{a \in D_i}$ は値域 D_i 上の確率分布 $\mathcal{D}_i$ である．次に，変数 x_i と x_j に関わるコスト関数 $f_{i,j} \in F$ に対して，変数 $\{\boldsymbol{\mu}_{i,j,a,b}\}_{a \in D_i, b \in D_j}$ を導入する．この変数は，各 $b \in D_j$ に対して，制約 $\sum_{a \in D_i} \boldsymbol{\mu}_{i,j,a,b} = \boldsymbol{x}_{j,b}$ をもち，各 $a \in D_i$ に対して，制約 $\sum_{b \in D_j} \boldsymbol{\mu}_{i,j,a,b} = \boldsymbol{x}_{i,a}$ をもつ．ここで，変数 $\boldsymbol{\mu}_{i,j,a,b}$ は変数 x_i と x_j にそれぞれ値 a と b を割り当てる確率を表す．つまり，変数集合 $\boldsymbol{\mu}_{i,j} := \{\boldsymbol{\mu}_{i,j,a,b}\}_{a \in D_i, b \in D_j}$ は値域 $D_i \times D_j$ 上の確率分布 $\mathcal{D}_{i,j}$ である．変数 x_i と x_j に対する周辺分布 $\mathcal{D}_{i,j}$ はそれぞれ分布 $\mathcal{D}_i$ と $\mathcal{D}_j$ に一致しなければならない．この LP の目的は，値の組 (a, b) が分布 $\boldsymbol{\mu}_{i,j}$ からサンプリングされるとき，コスト関数 $f_{i,j}(a, b)$ の総和を最小化することである．

まとめると，COP の LP 定式化は以下のとおりである．

$$\begin{aligned}
\min \sum_{f_{i,j} \in F} \boldsymbol{\mu}_{i,j,a,b} f_{i,j}(a,b), & \\
\text{subject to} \sum_{a \in D_i} \boldsymbol{x}_{i,a} = 1 & \quad \forall x_i \in X, \\
\sum_{a \in D_i} \boldsymbol{\mu}_{i,j,a,b} = \boldsymbol{x}_{j,b} & \quad \forall f_{i,j} \in F, b \in D_j, \\
\sum_{b \in D_j} \boldsymbol{\mu}_{i,j,a,b} = \boldsymbol{x}_{i,a} & \quad \forall f_{i,j} \in F, a \in D_i, \\
\boldsymbol{x}_{i,a} \geq 0 & \quad \forall x_i \in X, a \in D_i, \\
\boldsymbol{\mu}_{i,j,a,b} \geq 0 & \quad \forall f_{i,j} \in F, a \in D_i, b \in D_j.
\end{aligned}$$

一度 $\boldsymbol{x} := \{\boldsymbol{x}_{i,a}\}_{i \in [n], a \in D_i}$ の値が決定すると，$\boldsymbol{\mu} := \{\boldsymbol{\mu}_{i,j,a,b}\}_{i,j \in [n], a \in D_i, b \in D_j}$ の最適値を局所的に計算可能であることに注意されたい．事実，$\boldsymbol{\mu}_{i,j}$ は $\boldsymbol{x}_i$，$\boldsymbol{x}_j$，$f_{i,j}$ の情報を得るだけで計算できる．

劣勾配の計算

MW 法を適用するために，LP の解 $\boldsymbol{x}^t$ に対する劣勾配をコストに対応させる．DMW-LP では，エージェント i は点 $\boldsymbol{x}$ における f_i の劣勾配 $\boldsymbol{v}_i$ を計算する．f_i は $\boldsymbol{x}_i$ と $\{\boldsymbol{x}_j\}_{j \in N(i)}$ のみに依存するため，局所的に $\boldsymbol{v}_i$ を計算できる．

DMW-LP

DMW-LP は基本的に MW 法に従う．違いは，分散環境で動作する点である．DMW-LP では，エージェント i は重みベクトル $\boldsymbol{w}_i = (\boldsymbol{w}_{i,1}, \ldots, \boldsymbol{w}_{i,d_i})$ をもつ．ラウンド t における $\boldsymbol{x}_i^t$ は $\boldsymbol{x}_i^t = \boldsymbol{w}_i^t / \|\boldsymbol{w}_i^t\|_1$ で計算できる．また，意思決定 i のコストベクトルは，劣勾配の i 番目の部分に相当する．重みの更新方法は MW 法と同じ方法を用いる．まとめると，DMW-LP はアルゴリズム 2 のとおりである．特徴は，得られる解と LP の最適解との差を任意の小さい値にできる点である．

Algorithm 2 DMW-LP

Set $\eta \leftarrow \sqrt{\frac{\log d}{nT}}$
for each agent i **do**
  Set $\boldsymbol{w}_i^1 \leftarrow (1, 1, \ldots, 1)$ and $\boldsymbol{x}_i^1 \leftarrow (\frac{1}{d_i}, \frac{1}{d_i}, \ldots, \frac{1}{d_i})$.
  send $\boldsymbol{x}_i^1$ to each agent $j \in N(i)$.
for $t = 1$ to T **do**
  for each agent i **do**
    receive $\boldsymbol{x}_j^t$ from each agent $j \in N(i)$.
    Compute a subgradient $\boldsymbol{v}_i^t$ of f_i at $\boldsymbol{x}^t$.
    $\boldsymbol{w}_{i,a}^{t+1} \leftarrow \boldsymbol{w}_{i,a}^t(1 - \eta \boldsymbol{v}_{i,a}^t)$ for each $a \in D_i$.
    $\boldsymbol{x}_{i,a}^{t+1} \leftarrow \frac{\boldsymbol{w}_{i,a}^{t+1}}{\|\boldsymbol{w}_i^t\|_1}$ for each $a \in D_i$.
    send $\boldsymbol{x}_i^{t+1}$ to each agent $j \in N(i)$.
for each agent i **do**
  Assign x_i the value $\arg\max_{a \in D_i} \boldsymbol{x}'_{i,a}$, where $\boldsymbol{x}'_i = \frac{1}{T}\sum_{i=1}^T \boldsymbol{x}^t$.

最後に，ベクトル $\boldsymbol{x}' = \frac{1}{T}\sum_{i=1}^{T}\boldsymbol{x}^t$ を整数解に丸める．具体的には，各 $i \in [n]$ に対して，$x_i = \arg\max_{a\in D_i} \boldsymbol{x}'_{i,a}$ となる値を割り当てる．ここで，この方法は情報交換を一切必要としない点に注意されたい．

DMW-Game: ゲームに基づく解法

DMW-Game では，エージェントはコスト最小化ゲームに対するある種の均衡点を MW 法を用いて探索する．詳細は以下のとおりである．

コスト最小化ゲーム

まず，ゲーム理論から幾つかの概念を導入する．コスト最小化ゲームは以下の要素からなるゲームのことを指す.

- n 人からなる有限のプレイヤー，
- 各プレイヤー i に対する有限の意思決定集合 D_i,
- 各プレイヤー i に対するコスト関数 $f_i : D \to [0,1]$，ここで，$D = D_1 \times \cdots \times D_n$ とする．

以下の方法に沿ってコスト最小化ゲームをプレイする．この方法は，*no-regret dynamics* と呼ばれる．各ラウンド $t = 1, 2, ..., T$ において，各プレイヤーは以下のとおりに行動する．

- 各プレイヤー i は，no-regret 法を用いて同期的かつ独立に D_i 上の分布 $\mathcal{D}_i^t$ を選択する．
- 各プレイヤー i はコストベクトル $(\boldsymbol{c}_{i,1}^t, \ldots, \boldsymbol{c}_{i,1}^t)$ を受け取る．ここで，$\boldsymbol{c}_{i,a}^t$ は，他のプレイヤーが選択した分布に従ってプレイしたときの意思決定 a に対する期待コストである．つまり，$\boldsymbol{c}_{i,a}^t = \mathbf{E}_{x\sim\mathcal{D}^t}[f_i(x_1, \ldots, x_{i-1}, a, x_{i+1}, \ldots, x_n)]$ となる．ここで，$\mathcal{D}^t = \prod_{i\in[n]}\mathcal{D}_i^t$ とする．

no-regret dynamics で得られる解は coarse correlated equiliburium と呼ばれるある種の均衡点に収束する．

Theorem 2 (フォーク定理)**.** T ラウンド後，コスト最小化ゲームのプレイヤーは自身の意思決定に対して高々ϵ の *regret* しかもたない．$\mathcal{D}^t = \prod_{i=1}^{n}\mathcal{D}_i^t$ をラウンド t における分布とし，$\mathcal{D} = \frac{1}{T}\sum\mathcal{D}^t$ をそれまでに得られた分布の平均とすると，各プレイヤー i とその戦略 a_i に対して，

$$\mathop{\mathbf{E}}_{x\sim\mathcal{D}}[f_i(x)] \leq \mathop{\mathbf{E}}_{x\sim\mathcal{D}}[f_i(x_1, \ldots, x_{i-1}, a_i, x_{i+1}, \ldots, x_n)] + \epsilon.$$

を満たす分布 $\mathcal{D}$ を ϵ 近似 *coarse correlated equiliburium* と呼ぶ.

DMW-Game

DCOP は，D_i を i 番目の変数の値域，コスト関数 $f_i : D_i \to [0,1]$ を i 番目の変数 x_i に関わるコスト関数とすることでコスト最小化ゲームとみなせる．

no-regret dynamics において，MW 法を no-regret 法として用いると，アルゴリズム 3 が得られる．特徴は，得られる解と coarse correlated equiliburium との差を任意の小さい値にできる点である．

アルゴリズムの最後に，得られたベクトル $\boldsymbol{x}'$ を整数解に丸める必要がある．DMW-LP と同様に，各 $i \in [n]$ に対して，単純に $x_i = \arg\max_{a\in D_i}\boldsymbol{x}'_{i,a}$ とする．

Algorithm 3 DMW-Game

```
Set η = sqrt(log d_max / T).
for each agent i do
    Set w_i^1 ← (1, 1, ..., 1) and x_i^1 ← (1/d_i, 1/d_i, ..., 1/d_i).
    send x_i^1 to each agent j ∈ N(i).
for t = 1 to T do
    for each agent i do
        receive x_j^t from each agent j ∈ N(i).
        w_{i,a}^{t+1} ← w_{i,a}^t (1 − η f_i(a)) for each a ∈ D_i.
        x_{i,a}^{t+1} ← w_{i,a}^{t+1} / ||w_i^{t+1}||_1 for each a ∈ D_i.
        send x_i^{t+1} to each agent j ∈ N(i).
for each agent i do
    Assign x_i the value argmax_{a∈D_i} x'_{i,a}, where x'_i = (1/T) Σ_{i=1}^T x^t.
```

実験

提案した二つの DMW に対する解の質と効率性，スケーラビリティを確認する．そのために，DMW を二つの代表的な非厳密 DCOP アルゴリズムである MaxSum [Farinelli 08] と DeQED [Hatano 13] を用いて比較する．

この実験は，Intel Core-i7 3770@3.4GHz と 4GB メモリを搭載した Ubuntu サーバー上で実行した．DMW-LP と DMW-Game に関して，η の値をそれぞれ 0.04 と 0.5 に設定した．

DCOP 問題例

ランダムネットワーク，スケールフリーネットワーク，そして実世界の問題として会議日程調整問題の三種類の DCOP 問題例を生成した．最初の二つの問題例に関して，以下のとおりにネットワークを形成した．

ランダム 頂点数 n，密度 δ，枝数が $\lfloor\delta\binom{n}{2}\rfloor$ のネットワークを生成した．

スケールフリー Barabasi-Albert (BA) モデルを用いて頂点数 n のネットワークを生成した．ネットワークは，新しい頂点が追加される度に既存の頂点のうちの二点に枝を張ることにより生成される．枝の総数は $2(n-2)+1$ となる．

各トポロジーに対して 20 個のネットワークを生成した．このとき，部分ネットワークが存在しないことを確認したことに注意されたい．その後，生成した各ネットワークを基に，各変数（頂点）の値域サイズを 3，コスト関数（枝）のコストを $\{1, 2, ..., 10^5\}$ からランダムに選択し，COP 問題例を生成した．

会議日程調整問題に関しては，FRODO version2.11 のジェネレータをパラメータ”-PEAV -infinity 10^5 -maxCost 10^2 40 25 4 4”に設定し，20 問の問題例を生成した．

実行時間に対する解の質

まず，提案解法が，既存の解法と比較して効率的に良い質の解を求められることを実験的に示す．そのために，以下の 3 つの評価指標を考える．一つは，**解の質**で，求めたコストを DeQED_a から得られる下界で割ることより得られる．二つ目は，**サイクル数**で，DMW におけるラウンド数に対応する．三つ目は，**模擬実行時間**で，各サイクルの模擬実行時間の総和で表現される．また，あるサイクルにおける模擬実行時間は全エージェントのうち最長の実行時間を表す．

実験結果を図 1 に示す．ランダム，スケールフリーネットワークでは，DMW-Game は，解の質において他のアルゴリズ

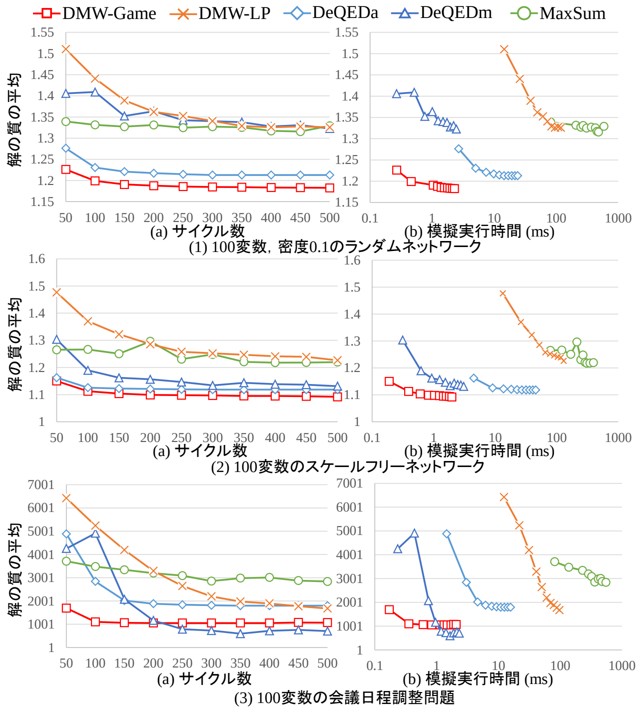

図 1: (1)100 変数，密度 0.1 のランダムネットワーク，(2)100 変数のスケールフリーネットワーク，(3)100 変数の会議日程調整問題における解の質の平均と模擬実行時間の平均

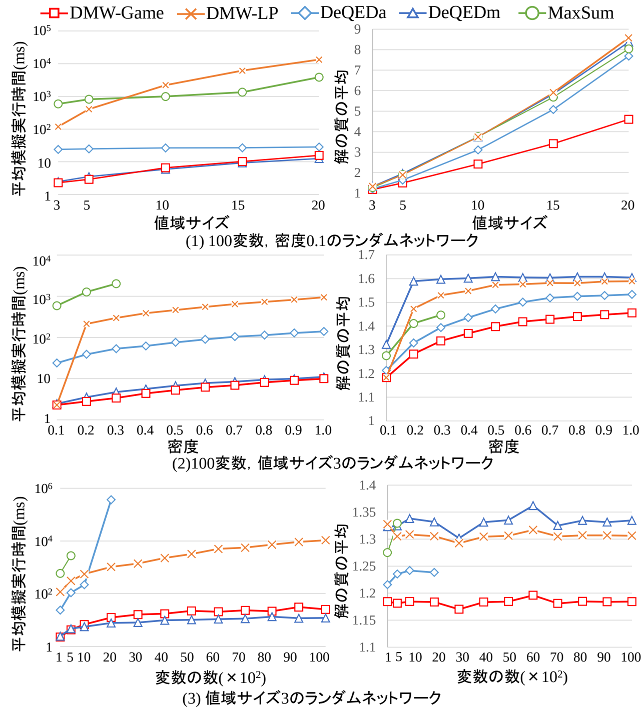

図 2: (1) 値域サイズ，(2) 密度，(3) 変数の数を変化させたときの解の質の平均と模擬実行時間の平均

ムより優れている．一方，会議日程調整問題では，二つ DMW は DeQED$_m$ と同等程度の解の質が得られている．この理由として，会議日程調整問題のコスト関数は実質ハード制約として見なされるため，DMW から得られた値を整数解に丸めるときにハード制約を満たすのに失敗していると考えられる．

二つの DMW を比較すると，どの DCOP 問題例においても DMW-Game は DMW-LP よりも良い質の解が得られていることがわかる．このことは，DMW-Game より得られた解とそれを丸めた後の整数解との差が DMW-LP のそれと比較して小さいことを意味する．さらに，DMW-Game は模擬実行時間に関してもより効率的に計算可能である．なぜなら，DMW-Game のエージェントは内積の計算を行うだけであるのに対して，DMW-LP では LP を解く必要があるためである．

スケーラビリティ

次に，DMW のスケーラビリティを評価する．そこで，ランダムネットワークの値域サイズ，密度，変数の数を変化させ実験する．このとき，密度に関しては 1 エージェント当たりのコスト関数の数の平均が保たれるように変化させた．

図 2 に実験結果を示す．図は，各設定に対して 500 サイクル時の解の質の平均と模擬実行時間の平均をプロットしたものである．メモリ制限のため，幾つかの巨大な問題例において，MaxSum と DeQED$_a$ が実行不可能であった．その理由として，これらの解法は計算コストや記憶容量が変数の数に対して増加するのに対して，DMW と DeQED$_m$ は値域サイズや隣接エージェントの数に対して線形にしか計算コストが増加しないことが考えられる．変数の数が大きいとき，DMW は最良の解を与えていることからも DMW は良いスケーラビリティをもつことが結論付けられる．

おわりに

本論文では，DCOP に対して乗算型重み更新法に基づく新しい非厳密解法を二つ提案した．一つは DMW-LP で線形緩和した問題を分散環境下で解く．その特徴は，得られる解と線形計画問題の最適解との差を任意の小さい値にできる点である．二つ目は，DMW-Game で，コスト最小化ゲームを扱う．その特徴は，得られる解と coarse correlated equilibrium との差を任意の小さい値にできる点である．

また，異なるトポロジー，コスト関数をもつ DCOP 問題例を用いて提案した 2 つの解法のスケーラビリティを実験的に示した．特に，DMW-Game は他の既存の解法と比較して良質な解を効率的に求められることが確かめられた．

参考文献

[Arora 12] Arora, S., Hazan, E., Kale, S.: The Multiplicative Weights Update Method: A Meta-Algorithm and Applications, Theory of Computing, pp. 121–164 (2012)

[Farinelli 08] Farinelli, A., Rogers A., Petcu A. and Jennings, N. R.: Decentralised Coordination of Low-Power Embedded Devices Using the Max-sum Algorithm, AAMAS-2008, pp. 639–646 (2008)

[Hatano 13] Hatano, D. and Hirayama, K.: DeQED: An Efficient Divide-and-Coordinate Algorithm for DCOP, IJCAI-2013, pp. 566–572 (2013)

[Léauté 09] Léauté, T., Ottens, B. and Szymanek, R.: FRODO 2.0: An Open-Source Framework for Distributed Constraint Optimization, DCR-09, pp. 160–164 (2009), http://liawww.epfl.ch/frodo/