Title	On designing optimal on-line algorithms for task systems against random players
Author(s)	山家, 明男; 櫻井, 幸一
Citation	数理解析研究所講究録 (1996), 950: 56-62
Issue Date	1996-05
URL	http://hdl.handle.net/2433/60341
Right	
Type	Departmental Bulletin Paper
Textversion	publisher

数理解析研究所講究録
950巻 1996年 56-62

On designing optimal on-line algorithms for task systems against random players

山家 明男 (Akio YANBE) 櫻井 幸一 (Kouichi SAKURAI)
九州大学 工学部 情報工学科 〒812-81 福岡市東区箱崎 6-10-1
Phone.092-642-4050 Fax.092-632-5204
{yanbe,sakurai}@csce.kyushu-u.ac.jp

1 はじめに

On-line Metrical Task System (S: システムの状態集合, T: 入力タスクの集合, A: タスク処理コスト行列, B: 状態遷移コスト行列, s_0: 初期状態) とは, 一単位時間に一つのタスクの受け付けとそのタスクの処理とシステムの状態の遷移を行うタスクシステムである. この時にかかるコストが行列 A, B で与えられ, それぞれ, 状態 i でタスク x を処理するコスト a_{ix}, 状態を i から j に遷移するコスト b_{ij} である. システムは, 連続して入ってくるタスクをなるべく小さなコストで処理していきたいのだが, タスクを処理するコストはそのときのシステムの状態によって違っているうえに, 状態遷移は未来のコストにも影響を与えるため, うまく状態遷移を行う必要がある. さらに, 未来の入力が事前にわからないないため, システムの状態遷移の決定はオンラインアルゴリズムとして設計する必要がある.

Borodin らは On-line Metrical Task System を Competitive Ratio の見地から解析を行っており, 状態数 n の任意の On-line Metrical Task System に対して Competitive Ratio が $8(n-1)$ のオンラインアルゴリズムを提案している [BLS92].

Zwick と Paterson は [ZP95] において, On-line Metrical Task System で adaptive adversary によるコストを最大にする最悪の入力は, Mean Payoff Game と呼ばれる二人ゲームにおける最適戦略として非決定性多項式時間で求められることを示した.

今回の研究では, Zwick らの方法を変更して, On-line Metrical Task System から別の形の Mean Payoff Game を構成し, 入力がランダムに入ってきた場合のシステムの最適なオンラインアルゴリズムを考えた. 結果として, 入力タスクがランダムに入ってくるときの, 平均コストを最小にする (最適な) オンラインアルゴリズムはある一つの状態にとどまって処理を行い続けるものであり, 多項式時間で計算できることを示した. さらに, Borodin らのアルゴリズムでランダム入力タスクを処理したときのコストと, この最適なアルゴリズムでのコストの比較を行った. これは言い替えると, 全ての入力に対する平均のコストの大きさを比較したことになる.

2 On-line Metrical Task System

On-line Metrical Task System [BLS92] とは, n 状態を持ち k 種類の要求を入力とするシステムである. 具体的に書くと On-line Metrical Task System は (S, T, A, B, s_0) であり, $S = \{1, 2, \cdots, n\}$ はシステムの状態集合, $T = \{1, 2, \cdots, k\}$ は入力タスク集合, A はタスク処理コスト行列で, A の要素 a_{ij} は状態 i でタスク j を処理するのに必要なコストである. B は状態遷移コスト行列で, B の要素 b_{ij} は状態を i から j に遷移させるのに必要なコストである. B は対称行列であり, 三角不等式 $(b_{ij} + b_{jk} \leq b_{ik})$ を満たすものとする. また, $s_0 \in S$ はシステムの初期状態である.

このシステムは 1 単位時間に次の三つの動作を行う.

step 1. システムが状態 i にあり, k 種類のタスクのうち一つ t を受けとる.

step 2. 状態 i でタスク t を処理する.

step 3. システムは状態 i から状態 j へ遷移する.

このうち, step 2 において状態 i でタスク t を処理するには a_{it} のコストがかかり, step 3 において状態 i から状態 j に遷移するには b_{ij} のコストがかかる. つまり, 1 単位時間で $a_{it} + b_{ij}$ のコストがかかることになる.

入力タスク列 I は $t_1 t_2 t_3 \ldots$ と表し, t_i が時間 i での入力とする. これに対して, スケジュール σ は s_0

s_1 s_2 s_3 ... と表し，状態 s_i でタスク t_i が処理されることになる．なお，入力タスクを入れる側 (タスクマスター) は入力するタスクがどの状態で処理されるのかを事前に知ることはできない．これはつまり，i 番目の入力タスク t_i はちょうど時間 i で入力され，t_i は $[i, i+1)$ の期間で処理が行われるという仮定によるものである．

システム側 (スケジューラ) はこの単位時間あたりの平均コストをなるべく低くおさえられるように，状態の遷移の仕方を考えなければならない．これはつまり，入力を読んで処理を行った後次に移る状態を決定するスケジュール σ を構成することであり，事前に入力がわからないことからオンラインアルゴリズムがこれを行うことになる．

σ を構成するアルゴリズムを X とし，一単位時間あたりの平均コストを $C_X(I)$ と書く．なお，I が事前に与えられた場合には，その最適なオフラインスケジュールを動的計画法によって多項式時間で求めることができる．入力列 I に対する最適オフラインスケジュールでのコストは $C_{off}(I)$ と表す．

2.1 トラバーサルアルゴリズム

Borodin らは [BLS92] で On-line Metirical Task System を Competitive Ratio の見地から解析を行い，状態数 n の任意の Metrical Task System に対して Competitive Ratio が $8(n-1)$ のオンラインアルゴリズムを提案している．このオンラインアルゴリズムはトラバーサルアルゴリズムと呼ばれ，トラバーサルアルゴリズムの入力タスク列 I に対するコストを C_τ とすると，

$$\max_I \frac{C_\tau(I)}{C_{off}(I)} < 8(n-1)$$

となるということである．

トラバーサル τ とは一続きの状態で，$\tau = s_0\ s_1\ s_2 \cdots s_l, (s_l = s_0), (s_i \neq s_{i+1}, 0 \leq i \leq l-1)$ と書かれる．トラバーサルアルゴリズムとはこの τ のことであり，s_i から s_{i+1} への状態の遷移は s_i においてタスクを処理するコストが $b_{s_i\ s_{i+1}}$ に達した時に起こる．

- トラバーサルアルゴリズムの構成方法 [BLS92]

  頂点数 n で各枝 (i,j) には b_{ij} の重みがつけられた完全グラフを考える．まず最小全域木 MST $= (S, E)$ を求める．次に MST のすべての枝 $(i,j) \in E$ について，その重みを $2^{p-1} < b_{ij} \leq 2^p$ を満たす 2^p に付けかえる．このグラフを MST' とする．MST' において重みが最大の枝の重みを 2^M とする．枝の重みが $b'_{ij} = 2^m$ の枝について一つの方向に 2^{M-m} 回ずつ通るように τ を構成する (枝を通る順番はどうでもよい)．

2.2 Mean Payoff Game との関係

Zwick と Paterson によって On-line Metrical Task System を Mean Payoff Game (付録 A) に変換し，タスクマスターを adaptive adversary とした場合の最悪入力を求める方法が提案されている [ZP95]．この最悪入力は，状態 s_i を利用して次の入力タスク t_{i+1} を決定するアルゴリズムとなるが，変換後の MPG の最適戦略がそれを示すようになっている．

ここで，与えられた On-line Metrical Task System $= \{S, T, A, B\}$ を Mean Payoff Game, $G = \{V_1, V_2, E\}$ に変換する方法は次のとおりである．

$V_1 = \{1, 2, \ldots, n\}$ とし，タスクマスターがプレイヤー I としてプレイする．$V_2 = \{(x,y) | 1 \leq x \leq n, 1 \leq y \leq k\}$ とし，スケジューラがプレイヤー II としてプレイする．

V_1 から V_2 への枝 $x \rightarrow (x,y)$, $x \in V_1$, $(x,y) \in V_2$ はシステムの状態が x で，タスクマスターがタスク y を入力したことを表す．V_2 から V_1 への枝 $(x,y) \rightarrow z$, $z \in V_1$, $(x,y) \in V_2$ は入力タスク y を処理する直前にシステムが状態を x から z へ換えることを表す．

各枝につけられる重み (コスト) は，$x \rightarrow (x,y)$ の形の枝すべてに 0，また $(x,y) \rightarrow z$ の形の枝には $a_{xz} + b_{zy}$ とする．

例 2.1 次の On-line Metrical Task System を考える．

$$S = \{1,2\}, T = \{1,2,3\}, A = \begin{bmatrix} 2 & 5 & 2 \\ 5 & 1 & 1 \end{bmatrix},$$

$$B = \begin{bmatrix} 0 & 3 \\ 3 & 0 \end{bmatrix}$$

上の方法で Mean Payoff Game に変換すると，図 1 のようになる．なお，$V_1 = \{1,2\}$,
$V_2 = \{(1,1),(1,2),(1,3),(2,1),(2,2),(2,3)\}$ である．

Mean Payoff Game では，プレイヤー I とプレイヤー II が交互に枝を選択していくルールになってい

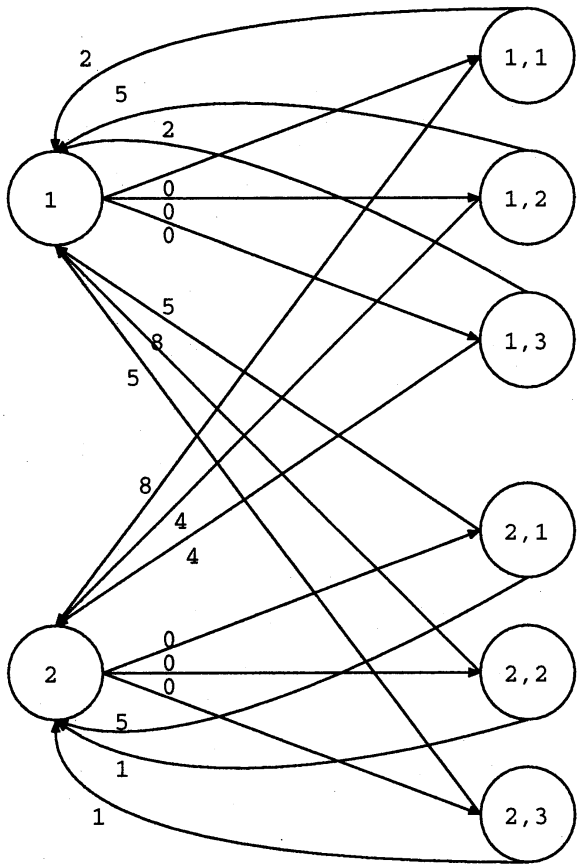

図 1: On-line Metrical Task System から Mean Payoff Game への変換

るが，プレイヤー I(タスクマスター) の選択する枝はどのタスクを入力するかということを表し，プレイヤー II(スケジューラ) の選択する枝は状態遷移を表している．この MPG 上で On-line Metrical Task System を考えてみると，プレイヤー I(タスクマスター) の最適戦略は各頂点で枝を一本選んだ形になるが，これはいまのシステムの状態から次の入力タスクを決定する adaptive な最悪入力になっている．また，プレイヤー II(スケジューラ) の最適戦略はシステムが入力タスクを一つ先読みできると仮定した時の，最適なオンラインアルゴリズムということになる．

3 An Optimal Schedule against a Random Taskmaster

本節では，On-line Metrical Task System におけるランダム入力を考える．この場合，On-line Metrical Task System を一人がランダムプレイヤーの Mean Payoff Game として考えることができるが，その変換方法は Zwick らの方法でのコストのつけ方を，$x \to (x,y)$ の形の枝すべてに 0，また $(x,y) \to z$ の形の枝には $a_{xz} + b_{xy}$ とするだけでよい．

例 3.1 例 2.1と同じタスクシステムで考えると，図 2のようになる．

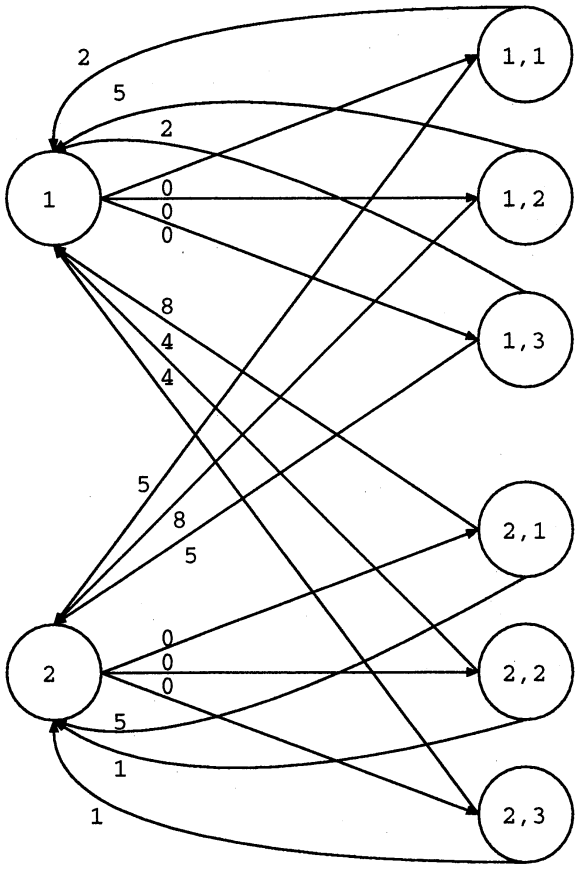

図 2: On-line Metrical Task System から Mean Payoff Game への変換

しかし，このままランダムプレイヤーと争う二人ゲームとして考えると，プレイヤー I をランダムプレイヤーとした Discounted Payoff Game として最適戦略を求めることになるが，以下の定理より直接求めた方がわかりやすい．

各時間で，入力タスク $\{1,2,\cdots,k\}$ がそれぞれ確率 $p_i, (1 \leq i \leq k)$ でランダムに発生するものとする．$p_1 = p_2 = \ldots = p_k$ とすると、すべての入力に対しての平均コストを考えることになる．

定理 3.2 On-line Metrical Task System において入力タスクがランダムに入ってくるとき，一つの状態にとどまって処理を行うことが，コストの期待値が最小になるという意味で最適なオンラインアルゴリズムである．

一つの状態にとどまるアルゴリズムを ONE–STATE アルゴリズムと呼ぶこととし，ONE–STATE アルゴリズムで入力列 I を処理した時のコストを $C_{one}(I)$ と書く．

ある状態 j で入力タスクを処理するコストの期待値は

$$E_j = a_{j1}p_1 + a_{j2}p_2 + \cdots + a_{jk}p_k$$

と計算される．よって，m 時間後の一単位時間あたりのコストの期待値 E は

$$E = \frac{\sum_{i=1}^{m}(E_{s_i} + b_{s_{i-1}\ s_i})}{m}$$

となる．これが最小となるのは，状態遷移をまったく行わないで，$\min_{1\le j\le n} E_j$ を満たす一つの状態 j で処理を続けるときである．このとき，最小値 E_{one} は

$$E_{one} = \min_{1\le j\le n} E_j$$

であり，これを満たす j が最適な状態である．また，

$$E_{one} = \mathop{\mathrm{AVE}}_{I} C_{one}(I)$$

と書け，これが最適 ONE–STATE アルゴリズムのコストの期待値となる．

系 3.3 ランダム入力タスクに対する最適 ONE–STATE アルゴリズムは多項式時間で計算可能である．

今回，ランダムな入力に対して，Borodin らのトラバーサルアルゴリズムのコストの期待値と定理 3.2の最適 ONE–STATE アルゴリズムのコストの期待値が，その比においてどれくらいになるか調べる．すなわち，

$$\frac{\mathop{\mathrm{AVE}}_{I_1} C_\tau(I_1)}{\mathop{\mathrm{AVE}}_{I_2} C_{one}(I_2)}$$

を求めるということである．

定理 3.4 ランダム入力に対してトラバーサルアルゴリズムで一単位時間にかかるコストの期待値 $E_\tau = \mathop{\mathrm{AVE}}_{I} C_\tau(I)$ は

$$2E_{one} \le E_\tau < 8(n-1)E_{one}$$

となる．また，$E_\tau = 2E_{one}$ となるのは $E_1 = E_2 = \cdots = E_n (= E_{one})$ のときに限る．

証明 トラバーサルアルゴリズムにランダム入力を入れた場合の一単位時間あたりのコストの期待値 $E_\tau = \mathop{\mathrm{AVE}}_{I} C_\tau(I)$ は次のように計算される．

一回のトラバーサルで重み $2^{m-1} < b_{ij} \le 2^m$ の枝 (i,j) を一方向に 2^{M-m} 回ずつ通るから，一回のトラバーサルでかかるコストの和 D_τ は

$$4(n-1)2^{M-1} < D_\tau \le 4(n-1)2^M$$

となる．

ここで $W_i, (1 \le i \le n)$ を，一回のトラバーサルにおいて状態 i でタスク処理にかかったコストの和とする．$deg(i)$ を MST における頂点 i の次数とすると，$1 \le i \le n$ に対して，

$$deg(i)2^{M-1} < W_i \le deg(i)2^M$$

である．

$$D_\tau = 2(W_1 + W_2 + \ldots + W_n)$$

だから，

$$4(n-1)2^{M-1} \le D_\tau < 4(n-1)2^M$$

である．

次に，一回のトラバーサルにかかる平均時間 T_τ を求める．

$$T_\tau = \frac{W_1}{E_1} + \frac{W_2}{E_2} + \ldots + \frac{W_n}{E_n}$$

$$\frac{W_i}{E_{one}} < T_\tau \le \frac{1}{E_{one}}(W_1 + \ldots + W_n)$$

$$\frac{2^{M-1}}{E_{one}} < T_\tau \le \frac{C_\tau}{2E_{one}}$$

よって，$E_\tau = \dfrac{D_\tau}{T_\tau}$ だから，

$$2E_{one} \le E_\tau < \frac{C_\tau}{2^{M-1}}E_{one}$$

$$2E_{one} \le E_\tau < 8(n-1)E_{one}$$

□

よって,

$$2 < \frac{\underset{I_1}{\mathrm{AVE}}\, C_\tau(I_1)}{\underset{I_2}{\mathrm{AVE}}\, C_{one}(I_2)} < 8(n-1)$$

である．これはすべての入力列に関するトラバーサルアルゴリズムとONE–STATEアルゴリズムの平均のコストの比であり，ONE–STATEアルゴリズムの方が2倍以上コストが低く押えられることがわかる．

4 終わりに

ランダム入力列に対するトラバーサルアルゴリズムや最適ONE–STATEアルゴリズムのコストの期待値は，それぞれ与えられたシステムごとに計算することができ，今回の結果につながった．次に，ランダム入力列に対するトラバーサルアルゴリズムのコストの期待値とオフラインアルゴリズムのコストの期待値との比を求めることを考えているが，オフラインアルゴリズムのコストの期待値を計算することはかなり難しいものと思われる．

ランダム入力列に対してさらなる解析を行うことによって，最終的にはトラバーサルアルゴリズムのCompetitive Ratioの平均の値 $\underset{I}{\mathrm{AVE}}\, \frac{C_\tau(I)}{C_{off}(I)}$ を，与えられるシステムごとに計算できるようになることが望ましい．

また，Mean Payoff Gameを利用してシステムのオンラインアルゴリズムを設計することも検討課題である．

参考文献

[BLS92] Borodin, A., Linial, N. and Saks, M.E., "An optimal on-line algorithm for metrical task system," *Journal of the Association for Computing Machinery* **39** (1992) pp. 745–763.

[CHPZ95] Cole, R., Hariharan, R., Paterson, M. and Zwick, U., "Tighter lower bounds on the exact complexity of string matching,"*SIAM Journal on Computing* **24** (1995) pp. 30–45.

[Con92] Condon, A., "The Complexity of Stochastic Games," *Information and Computation* **96** (1992) pp. 203–224.

[EM79] Ehrenfeucht, A. and Mycielski, J., "Positional strategies for mean payoff games," *International Journal of Game Theory* **8** (1979) pp. 109–113.

[GKK88] Gurvich, V.A., Karzanov, A.V. and Khachivan, L.G., "Cyclic games and an algorithm to find minimax cycle means in directed graph," *USSR Comput. Maths. Math. Phys.* **28** (1988) pp. 85–91.

[Kar78] Karp, R.M., "A characterization of the minimum cycle mean in a digraph," *Discrete Mathematics* **23** (1978) pp. 309–311.

[KL93] Karzanov, A.V. and Lebedev, V.N., "Cyclical games with prohibitions," *Mathematical Programming* **60** (1993) pp. 277–293.

[YS96a] Yanbe, A. and Sakurai, K., "A short certificate of the number of universal optimal strategies for stopping simple stochastic games," *IPL* **57** (1996) pp. 17-24.

[YS96b] Yanbe, A. and Sakurai, K., "On the complexity of computational problems associated with simple stochastic games,"*to appear COCOON'96.*

[ZP95] Zwick, U. and Paterson, M., "The complexity of mean payoff games on graphs," *ECCC Reports Series 1995* **TR95-040.**

A Mean Payoff Game

Mean Payoff Gameは有向グラフ $G = (V_1, V_2, E)$, 枝の重み関数 $w : E \to R$ およびスタート頂点 $a_0 \in V_1 \cup V_2$, で与えられる．有向グラフではどの頂点も枝の次数は1以上である．

この有向グラフ上でのプレイは，まずスタート頂点に駒が置かれ，この駒を二人のプレイヤーが次のように動かしていく．V_1 頂点上に駒がある時にはプレイ

ヤーIが，V_2 頂点上に駒がある時にはプレイヤーIIが駒を動かす．この時通った枝を $e_1, e_2, \cdots$ とすると，プレイヤーIは利得 $\lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} w(e_i)$ を大きくすることを目的とし，プレイヤーIIは小さくすることを目的とする．

このMPGでは最適値 $\mathcal{V}$ が存在し，プレイヤーIには少なくとも $\mathcal{V}$ 以上の利得を保証する戦略があり，プレイヤーIIには利得が多くて $\mathcal{V}$ 以下になることを保証する戦略がある [EM79]．この時の戦略を最適戦略という．なお，この戦略は，各頂点で枝を一本選び常にその枝に沿って駒を動かす形の戦略になっている．

MPGの最適値や最適戦略を求める多項式アルゴリズムは見つかっておらず，指数時間アルゴリズムがGurvichらによって提案されている [GKK88]．

B Discounted Payoff Game

Discounted Payoff Game(DPG) は Mean Payoff Game の discounted version である．Discounted Payoff Game はそれ自体としても興味深いゲームであるとともに，Mean Payoff Game を Simple Stochastic Game(SSG) に変換する時に中渡しとなるゲームである．

MPGと同様に二人のプレイヤーが駒を動かしていくが，利得の計算がMPGとは少し違っている．λ を $0 < \lambda < 1$ の実数とする．プレイヤーによって i 番目に選ばれた枝 e_i の重みには $(1-\lambda)\lambda^{i-1}$ がかけられ，ゲームの利得は $(1-\lambda)\sum_{i=0}^{\infty} \lambda^{i-1} w(e_i)$ となる．プレイヤーIはこの利得を大きくすることが目的で，プレイヤーIIは利得を小さくすることが目的である．λ は *discounting factor* と呼ばれる．

DPGは有向グラフ $G = (V_1, V_2, E)$，枝の重み関数 $w : E \to R$，スタート頂点 $i \in V_1 \cup V_2$，および実数 λ で与えられる．有向グラフではどの頂点も枝の次数は1以上である．$V = V_1 \cup V_2 = \{1, 2, \cdots, n\}$ とする．$x_i (= x_i(\lambda))$ は頂点 i からスタートする時の discounted game の利得とする．

このDPGの最適値や最適戦略の存在は次の定理によって示されている．

定理 B.1 ([ZP95]) discounted payoff game の最適値 $\overline{x} = \{x_1, x_2, \cdots, x_n\}$ はつぎの方程式の唯一の解である．

$$x_i = \begin{cases} \max_{(i,j)\in E}\{(1-\lambda)w_{ij} + \lambda x_j\} & \text{if } i \in V_1, \\ \min_{(i,j)\in E}\{(1-\lambda)w_{ij} + \lambda x_j\} & \text{if } i \in V_2. \end{cases}$$

discounting factor が λ である DPG の最適値を $\mathcal{V}(\lambda)$ とする．λ を1に近付けると $\mathcal{V}(\lambda)$ は Mean Payoff Game の最適値 V に近付くので，Mean Payoff Game は Discounted Payoff Game に帰着できることになる [ZP95]．

C 確率 discounted Payoff Games

確率 discounted payoff game(確率 DPG) は DPG にランダムプレイヤーを追加したものである．確率 DPG は有向グラフ $G = (V_1, V_2, V_3, E)$，枝の重み関数 $w : E \to R$，スタート頂点 $i \in V_1 \cup V_2 \cup V_3$，および実数 λ で与えられる．V_1 の頂点上に駒がある時にはプレイヤーIが駒を動かし，V_2 の頂点上に駒がある時にはプレイヤーIIが駒を動かす．さらに，V_3 の頂点 i から出ている各枝 (i,j) には確率 r_{ij} がつけてあり，頂点 i に駒がある時にはこの確率にしたがって駒が動くことになる．各頂点 $i \in V_3$ について $\sum_{(i,j)\in E} r_{ij} = 1$ である．

DPGと同様に，プレイヤーによって i 番目に選ばれた枝 e_i の重みには $(1-\lambda)\lambda^i$ がかけられ，ゲームの利得は $(1-\lambda)\sum_{i=0}^{\infty} \lambda^i w(e_i)$ となる．しかし，ランダムプレイヤーが存在するため，この利得の期待値を確率 DPG の利得とする．プレイヤーIはこの利得を大きくすることが目的で，プレイヤーIIは利得を小さくすることが目的である．

D 確率 Discounted Payoff Game の最適値

定理 D.1 確率 discounted payoff game $G = (V_1, V_2, V_3, E)$ の最適値 $\overline{x} = \{x_1, x_2, \cdots, x_n\}$ はつぎの方程式の唯一の解である．

$$x_i = \begin{cases} max_{(i,j)\in E}\{(1-\lambda)w_{ij} + \lambda x_j\} & \text{if } i \in V_1, \\ min_{(i,j)\in E}\{(1-\lambda)w_{ij} + \lambda x_j\} & \text{if } i \in V_2, \\ \sum_{(i,j)\in E} r_{ij}\{(1-\lambda)w_{ij} + \lambda x_j\} & \text{if } i \in V_3. \end{cases}$$

証明：$\mathcal{F}$ を，任意のベクトル $\overline{x}$ を引数とし $\overline{y}$ を返す次のような関数とする．

$$y_i = \begin{cases} \max_{(i,j)\in E}\{(1-\lambda)w_{ij} + \lambda x_j\} & \text{if } i \in V_1, \\ \min_{(i,j)\in E}\{(1-\lambda)w_{ij} + \lambda x_j\} & \text{if } i \in V_2, \\ \sum_{(i,j)\in E} r_{ij}\{(1-\lambda)w_{ij} + \lambda x_j\} & \text{if } i \in V_3. \end{cases}$$

まず，$\overline{x} = \mathcal{F}(\overline{x})$ となるような $\overline{x}$ が存在することを示す．$||\overline{v}|| = max_i\{|v_i|\}$ とする (最大ノルム)．

$$\forall \overline{u}, \overline{v},\ ||\mathcal{F}(\overline{u}) - \mathcal{F}(\overline{v})|| \leq \lambda||\overline{u} - \overline{v}||$$

よって，$0 < \lambda < 1$ より $\mathcal{F}$ は最大ノルムを小さくするような関数である．ゆえに，$\overline{x} = \lim_{n\to\infty} \mathcal{F}^n(\overline{0})$ が存在し，これが $\overline{x} = \mathcal{F}(\overline{x})$ の唯一の解となる．

$i \in V_1$ のとき

[1] $\mathcal{F}(u_i) = (1-\lambda)w_{ik} + \lambda u_k, \mathcal{F}(v_i) = (1-\lambda)w_{ik} + \lambda v_k$ のとき

$$\mathcal{F}(u_i) - \mathcal{F}(v_i) = \lambda(u_k - v_k)$$

$|u_k - v_k| \leq ||\overline{u} - \overline{v}||$ より

$$|\mathcal{F}(u_i) - \mathcal{F}(v_i)| \leq \lambda||\overline{u} - \overline{v}||$$

[2] $\mathcal{F}(u_i) = (1-\lambda)w_{ij} + \lambda u_j, \mathcal{F}(v_i) = (1-\lambda)w_{ik} + \lambda v_k, j \neq k$ のとき

$$\mathcal{F}(u_i) = (1-\lambda)w_{ij} + \lambda u_j \geq (1-\lambda)w_{ik} + \lambda u_k$$

$$\mathcal{F}(v_i) = (1-\lambda)w_{ik} + \lambda v_k \geq (1-\lambda)w_{ij} + \lambda u_j$$

$$\mathcal{F}(u_i) - \mathcal{F}(v_i) \geq \lambda(u_j - v_j)$$

$$\mathcal{F}(u_i) - \mathcal{F}(v_i) \leq \lambda(u_k - v_k)$$

よって

$$\lambda(u_k - v_k) \leq \mathcal{F}(u_i) - \mathcal{F}(v_i) \leq \lambda(u_j - v_j)$$

$|u_k - v_k| \leq ||\overline{u} - \overline{v}||, |u_j - v_j| \leq ||\overline{u} - \overline{v}||$ より

$$|\mathcal{F}(u_i) - \mathcal{F}(v_i)| \leq \lambda||\overline{u} - \overline{v}||$$

$i \in V_3$ のとき

$$\mathcal{F}(u_i) = r_1\{(1-\lambda)w_1 + \lambda u_{j1}\} + \cdots + r_p\{(1-\lambda)w_p + \lambda u_{jp}\}$$

$$\mathcal{F}(v_i) = r_1\{(1-\lambda)w_1 + \lambda v_{j1}\} + \cdots + r_p\{(1-\lambda)w_p + \lambda v_{jp}\}$$

よって

$$\mathcal{F}(u_i) - \mathcal{F}(v_i) = \lambda\{r_1(u_{j1} - v_{j1}) + \cdots + r_p(u_{jp} - v_{jp})\}$$

$|r_1(u_{j1} - v_{j1}) + \cdots + r_p(u_{jp} - v_{jp})| \leq ||\overline{u} - \overline{v}||$ だから

$$\mathcal{F}(u_i) - \mathcal{F}(v_i) \leq \lambda||\overline{u} - \overline{v}||$$

$\overline{x}$ を $\overline{x} = \mathcal{F}(\overline{x})$ の唯一の解とする．プレイヤーIは戦略として各頂点 $i \in V_1$ で $(1-\lambda)w_{ij} + \lambda x_j$ を最大とするような頂点 j への枝 (i,j) を選べば，各頂点 i からゲームがスタートする時の利得の期待値を少なくとも x_i 以上に保てることは明らかである．同様に，プレイヤーIIの戦略として各頂点 $i \in V_2$ で $(1-\lambda)w_{ij} + \lambda x_j$ を最小とするような頂点 j への枝 (i,j) を選べば，各頂点 i からゲームがスタートする時の利得の期待値を多くて x_i 以下に保てる．つまり，頂点 i からスタートするゲームの最適値は x_i となる． □

この定理から，二人のプレイヤーは各頂点で枝を一本選んで常にその枝に沿って駒を動かすような純粋戦略の形で最適戦略を持つことがわかる．

また，プレイヤーIIをランダムプレイヤーとしたDPG，プレイヤーIをランダムプレイヤーとしたDPGにおいても最適値，最適戦略が存在することがわかる．

DPGは多項式時間でSimple Stochastic Game [Con92]に変換できる [ZP95]が，この変換方法で，プレイヤーIIをランダムプレイヤーとしたDPGはMAX & AVE SSGに変換され，プレイヤーIをランダムプレイヤーとしたDPGはMIN & AVE SSGに変換される．MAX & AVE SSGとMIN & AVE SSGは多項式時間で最適値および最適戦略が計算できるので [Con92]，次の定理がいえる．

定理 D.2 プレイヤーI(またはII)をランダムプレイヤーとしたDPGの最適値および最適戦略は多項式時間で計算できる．

よって次の定理がいえる．

定理 D.3 確率Discounted Payoff Gameにおいて相手の戦略を知っている場合に，その戦略に対して最適な戦略を多項式時間で計算することができる．

確率DPGに関して自然に考えられる判定問題は次のようになる．確率DPGとある値 k が与えられたときに，確率DPGの最適値が k より大きいか? という問題である．

定理 D.4 確率DPGにおける判定問題は NP ∩ co-NP である．