ベイズ推定に基づくタスク順序付け

斉 藤 和 巳[†] 中 野 良 平[†]

本論文では，タスク順序付け問題，および，これを一般化した論理式の真偽値判定問題において，タスク成功確率をベイズ推定で求める方法が最尤推定を用いる方法より優ることを示す．タスク順序付け問題とは，実行コストが既知であるタスクが列をなし，どれかのタスクが成功すれば処理が完了するとき，処理が完了するまでの期待コストを最小にするタスク列を求める問題である．このとき，最尤推定してタスク列を求めても，事例が少ない段階では，最尤推定値の信頼性が極めて低いので，適切なタスク列を生成できない．一方，ベイズ推定してタスク列を求めれば，生成したタスク列の平均期待コストを最小にできることを示す．計算機実験で，ベイズ推定を用いる方法が，ランダムなタスク列や最尤推定してタスク列を求める方法と比較して，任意の事例数において，期待コストが最小のタスク列を生成し，最小コストにも速く近付くことを確認した．また，タスク順序付け問題を一般化すると μ-式というクラスに属す論理式の真偽値判定問題となる．その問題はタスク順序付け問題の解法を再帰的に適用して解けることを示す．事例実験で，ベイズ推定を用いる方法の優秀性を示す．

Task Sequencing Based on Bayesian Estimation

KAZUMI SAITO[†] and RYOHEI NAKANO[†]

A given finite set of tasks, having known costs for their trials, are assumed to be performed sequentially until any one of tasks successes. The task sequencing problem is to determine an optimal task sequence having the minimal expected cost. In the case of known success probabilities of all tasks, we show that the optimal task sequence can be generated easily when the success events of tasks are mutually exclusive or independent. In the case of unknown success probabilities of all tasks, since the maximum likelihood estimation is not reliable when the number of available facts is small, it is likely to generate poor task sequences. Whereas we show that a method based on the Bayesian estimation minimizes average expected costs under the assumption that a priori knowledge about the success probabilities is unknown. The experiments show that the expected costs of the task sequences generated by the Bayesian estimation are less than those of random task sequences or the task sequences generated by the maximum likelihood estimation. Moreover, the task sequencing problem is generalized to the truth assignment problem of formulae belonging to μ-formula, and the experiments show that the Bayesian estimation works well also on this problem.

1. はじめに

知能の基本的な特徴の1つは，過去の経験に基づいて，外界から与えられる仕事を問題解決する性能（正答率や効率）を適応的に改善できることであり，そのメカニズムの研究は，人工知能研究の重要テーマの1つである．

適応メカニズムの中心技術に，事例の出現やタスクの成功などに関する確率推定がある．最も自然な推定法は最尤推定であるが，与えられた事例に対する対数尤度を最大にするので，利用できる事例が少ない段階では，事例へのオーバーフィットが起こり，推定値の信頼性が極めて低くなると予想される．事例が少ないときにも有効な推定法として，事前確率に関する知識がなければ一様分布を仮定するラプラスの法則（Laplace's law）に基づいたベイズ推定法が提案され[4]，分類問題に現れる2項分布の確率推定への応用では，その有効性が検討されている[1]．しかるに，確率推定は幅広い分野で利用可能であり，多項分布の場合も含め，より多くの問題での有効性の検証が望まれる．

確率推定しながら問題解決する基本的なやり方の1つに，タスク順序付け（task sequencing）問題[3]がある．すなわち，実行コストが既知であるタスクが列をなし，新たに出現する事例に対して，列の先頭のタスクから順にどれかのタスクが成功するまで処理が試みられるとき，処理を完了するまでの期待コストを最小

† NTT コミュニケーション科学研究所
NTT Communication Science Laboratories

にするタスク列を求める問題である．タスク成功確率が既知で成功事象に相関がなければ，各タスクの成功確率とコストの比に基づいて，タスクを降順に並べれば，期待コストは最小になることが知られている[3),7),8)]．しかし，一般の現実問題では，タスク成功確率は未知であり，過去の事例から推定しなければならない．

タスク順序付け問題には，さまざまな現実問題への応用が考えられる[3),7),8)]．例えば，交換機や計算機の故障診断では，故障が起これば，保守者は適切な診断を順に行い，コスト (e.g.時間) 最小で故障箇所を発見することが望まれる．多重故障確率が非常に小さいとき(タスク群が排反のとき)，または，故障箇所に相関がなければ(タスク群が独立のとき)，診断の期待コストを最小にする診断列を求めることは，タスク順序付け問題となる．また，一般の探索問題では，あるノードが最適解へ導く確率，および，そのノードから解に至るまでのコストが推定されるとき，タスク順序付け問題と同じ評価尺度で最良優先探索すれば，最適な探索となることが証明されている[7)]．

本論文では，まず，タスク順序付け問題を説明した後，タスク成功確率が既知の場合の最適タスク列生成法について述べる．次いで，タスク成功確率が未知の場合に，最尤推定とベイズ推定でのタスク列生成法を示し，それぞれの方法についての考察と実験結果について報告する．さらに，タスク順序付け問題を論理式の真偽値判定問題として一般化し，その解法を示した後，確率推定法の違いに関する実験結果を報告する．

2. タスク順序付け問題

タスク集合を $\{T_1, \cdots, T_n\}$，タスク T_i の実行コストを c_i，任意のタスク列を $\sigma=(T_{\sigma(1)}, \cdots, T_{\sigma(n)})$ とする．また，各事例に対しては，$T_{\sigma(1)}$ から順に，どれかのタスクが成功するまで，処理が試みられるとする．タスク T_i が成功する事象を s_i，失敗する事象を $\neg s_i$ とすれば，タスク $T_{\sigma(i)}$ が実行されるのは，それ以前に実行したタスクがすべて失敗したときであり，事例の処理が完了するまでの期待コスト $E(\sigma)$ は：

$$E(\sigma)=\sum_{i=1}^{n} q_{\sigma(i)} c_{\sigma(i)}$$

で定義できる．ここで，確率 $q_{\sigma(i)}$ は：

$$q_{\sigma(i)}=\begin{cases}1 & i=1\\ P(\neg s_{\sigma(1)}\wedge\cdots\wedge\neg s_{\sigma(i-1)}) & i>1\end{cases}$$

で定義され，タスク $T_{\sigma(i)}$ が実行される確率である．タスク順序付け問題とは，期待コスト $E(\sigma)$ を最小にするタスク列 σ を求める問題である．

確率 $q_{\sigma(i)}$ を求めるには，一般に，任意のタスクに関する条件付き確率を考慮しなければならないが，多くの場合，現実的な仮定を導入できる．すなわち，各タスクの成功事象は，互いに排反 (exclusive) であるか，または，互いに独立 (independent) であると仮定する．ここで，事象 $e_1, \cdots, e_n$ が排反とは，

$$P(e_1\vee\cdots\vee e_n)=P(e_1)+\cdots+P(e_n)$$

であり，事象 $e_1, \cdots, e_n$ が独立とは，

$$P(e_1\wedge\cdots\wedge e_n)=P(e_1)\times\cdots\times P(e_n)$$

である．独立ケースについては，従来より広く研究され，さまざまな応用が知られている[3),7),8)]．一方，排反ケースについては，あまり研究されていないが，この例には，多重故障確率が極めて小さい故障診断問題があり，応用分野の広い重要なケースであると考える．以下では，タスク T_i の成功確率を $P(s_i)=p_i$ とする．

タスクの成功事象が互いに排反であり，全成功確率の和が 1 であるときには，確率 $q_{\sigma(i)}$ は：

$$q_{\sigma(i)}=1-\sum_{j=1}^{i-1} p_{\sigma(j)} = \sum_{j=i}^{n} p_{\sigma(j)}$$

となる．従って，期待コスト $E(\sigma)$ は：

$$E(\sigma)=\sum_{i=1}^{n}\left(\sum_{j=i}^{n} p_{\sigma(j)}\right)c_{\sigma(i)}$$

となる．一方，タスクの成功事象が互いに独立であるときには，その否定も互いに独立となるので，$i>1$ ならば，確率 $q_{\sigma(i)}$ は：

$$q_{\sigma(i)}=\prod_{j=1}^{i-1}(1-p_{\sigma(j)})$$

となる．従って，期待コスト $E(\sigma)$ は：

$$E(\sigma)=\sum_{i=1}^{n}\left(\prod_{j=0}^{i-1}\left(1-p_{\sigma(j)}\right)\right)c_{\sigma(i)}$$

となる．ただし，$p_{\sigma(0)}=p_0=0$ とする．

3. 最適タスク列

各タスクの成功確率 p_i を既知とすれば，最適タスク列は容易に得られる．

定理 1 タスクの成功事象が排反か独立であり，p_i 既知の場合は，タスクを p_i/c_i の大きい順に並べれば，期待コストが最小となる．

証明 タスク列 $\sigma(i)=i$ において，$p_i/c_i\geq p_{i+1}/c_{i+1}$ が成り立つとする．まず，タスクの成功事象が互いに排反のケースでは，期待コスト $E(\sigma)$ は：

$$E(\sigma)=\sum_{i=1}^{n}\left(\sum_{j=i}^{n} p_j\right)c_i$$

である．ここで，タスク列 σ の k と $k+1$ 番目のタスクを入れ換えたタスク列 σ' を考えれば，その期待コストの差は：

$$E(\sigma')-E(\sigma)=c_{k+1}p_k-c_kp_{k+1}$$
$$=c_kc_{k+1}\left(\frac{p_k}{c_k}-\frac{p_{k+1}}{c_{k+1}}\right)$$
$$\geq 0$$

となる．従って，タスク列 σ に対して任意の置換を施せば，期待コストは同じか増加する．

次に，タスクの成功事象が互いに独立のケースでは，期待コスト $E(\sigma)$ は：

$$E(\sigma)=\sum_{i=1}^{n}\left(\prod_{j=0}^{i-1}(1-p_j)\right)c_i$$

である．同様に，k と $k+1$ 番目のタスクを入れ換えたタスク列 σ' を考えれば，その期待コストの差は：

$$E(\sigma')-E(\sigma)=(c_{k+1}p_k-c_kp_{k+1})\prod_{j=0}^{k-1}(1-p_j)$$
$$\geq 0$$

となる．従って，このケースでも，タスク列 σ に対して任意の置換を施せば，期待コストは同じか増加する．

そこで，任意のタスク列 τ に対して，$E(\tau)\geq E(\sigma)$ となることを示す．このため，隣合うタスクの入れ換えによる σ から τ への変換を考える．まず，τ での n 番目（最後）のタスクを T_z とする．$T_z=T_n$ でなければ，σ において T_z をすぐ後のタスクと入れ換える操作を繰り返すことにより，T_z が n 番目となるタスク列 $\sigma^{(n)}$ を作れる．仮定より，$p_i/c_i\geq p_{i+1}/c_{i+1}$ であるので，上記の結果 $(E(\sigma')\geq E(\sigma))$ を考慮すれば，$E(\sigma^{(n)})\geq E(\sigma)$ となる．次に，τ での $n-1$ 番目のタスクを T_y とする．同様にして，必要ならば $\sigma^{(n)}$ において T_y をすぐ後のタスクと入れ換える操作を繰り返すことにより，T_y が $n-1$ 番目となるタスク列 $\sigma^{(n-1)}$ を作れる．ここで，$\sigma^{(n)}$ での始めから $n-1$ 番目までのタスクにおいては，任意の隣合うタスクのペア $[T_i, T_j]$ に対して，$p_i/c_i\geq p_j/c_j$ が成り立つので，$E(\sigma^{(n-1)})\geq E(\sigma^{(n)})$ となる．以下，これらの操作を順次繰り返せば，$\tau=\sigma^{(1)}$，$E(\sigma^{(1)})\geq\cdots\geq E(\sigma)$ であり，$E(\tau)\geq E(\sigma)$ となる．□

タスクの成功事象が互いに排反，または，独立であると仮定しても，最適タスク列を生成する評価尺度は同じものとなるが，期待コストの差に関しては，異なる性質を持つ．すなわち，前者の場合には，期待コストの差は，入れ換える2つのタスクの成功確率とコストだけに依存する．一方，後者の場合には，$k\geq 2$ では，期待コストの差は，入れ換える2つのタスクの成功確率とコストだけでなく，k 以前に現れるタスクの成功確率にも依存する．なお，後者の結果は，多くの研究者により，独立に発見されている[3),7),8)]．

4．確率推定に基づくタスク順序付け

一般に，タスク成功確率 p_i は未知なので，事例より推定しなければならない．代表的な確率推定法には，最尤推定法とベイズ推定法がある．以下では，それぞれの方法について述べる．また，次章では，それぞれの方法を用いた実験結果について報告する．

4.1　最尤推定法

まず，タスクの成功事象が互いに排反であるときには，タスクの成功は多項分布に従うので，総事例数を m，タスク T_i が成功した事例数を m_i とすれば，タスク成功確率は：

$$\hat{p}_i=\frac{m_i}{m} \tag{1}$$

により最尤推定できる．一方，タスクの成功事象が互いに独立であるときには，n 個の二項分布の集まりとみなせるので，同様に，(1)式により，最尤推定できる．従って，定理1より，タスクを m_i/c_i の大きい順に並べれば，タスク成功確率の最尤推定値 $\{\hat{p}_i\}$ に対する期待コストが最小になる．以下では，この評価尺度に基づいてタスク列を生成する方法を最尤推定法と呼ぶ．

大数の法則より，非常に多くの事例が与えられれば，$p_i\approx\hat{p}_i$ となるので，最小に近い期待コストで新たに現れる事例を処理できる．しかし，医療などへの応用では，タスク実行コストが非常に高いケースが考えられ，利用できる事例が少ない段階でも，できるだけ期待コストが小さいタスク列を生成する方法が望まれる．しかるに，最尤推定法は少ない事例に対して優れたタスク列を与える保証はない．また，最尤推定してタスク列を求めても，一般に，期待コストが改善されない場合がある（詳細は後述する）．

4.2　ベイズ推定法

ラプラスの法則とは，タスク成功確率に関する事前知識がなければ，それらに一様分布を仮定することである．次の定理より，ラプラスの法則に基づくベイズ推定（以下では，単にベイズ推定と呼ぶ）には，望ましい性質を示すことができる．

定理2　タスクの成功事象が排反か独立のとき，ベイズ推定に基づいて，タスクを $(m_i+1)/c_i$ の大きい順に並べれば，生成したタスク列の平均期待コストが最小になる．

証明　付録参照．□

以下では，この評価尺度に基づいてタスク列を生成する方法をベイズ推定法と呼ぶ．

最尤推定法の評価尺度 m_i/c_i の代わりに，ベイズ推定法では $(m_i+1)/c_i$ を用いることの妥当性について考察する．タスクの成功事象が互いに排反の場合には，ベイズ推定に基づけば，タスク成功確率は：

$$\hat{p}_i = \frac{m_i + 1}{m + n}$$

により推定されることになる．まず，初期タスク列に関しては，$m = m_i = 0$ なので，$\hat{p}_i = 1/n$ となり，等確率になる．一方，最尤推定法では確率が定まらない．タスク成功確率が未知のときには，それらをすべて等しいと仮定するのは極めて自然であり，ベイズ推定法により合理的な推定値が得られる．

次に，事例が少ない段階では，最尤推定法では，望ましくないタスクのペアの入れ換えが起こり得ることを示す．いま，$p_i/c_i > p_j/c_j$ であり，c_i は c_j より十分小さいが，p_j は p_i より若干大きいとする．まず，初期タスク列を $\hat{p}_i = \hat{p}_j$ として定めると，それは望ましい順番となっている．その後，$m_i = 0$，$m_j = 1$ となる確率は，$m_i = 1$，$m_j = 0$ が起こる確率よりも大きい．$m_i = 0$，$m_j = 1$ が起こると，最尤推定法では，望ましくないタスクのペアの入れ換えが起こる．従って，c_i が c_j より十分小さく，事例が少ないときは，タスクの入れ換えを行うべきではない．この問題に対処するには，コストだけに基づいた尺度 $(1/c_i)$，および，推定確率とコストを用いた尺度 (m_i/c_i) の間でのトレードオフを考えなければならない．ベイズ推定法では，両者の単純な和を尺度にすべきことを示している．

最後に，事例が十分に多い段階になれば，最尤推定値の信頼性は高くなる．一方，一般に m_i と m の値は 1 に比較して十分大きな値となる．従って，この段階では，1 を加えることが無視できるようになり，最尤推定法とベイズ推定法は，ほぼ同じタスク列を生成することになる．

一方，タスクの成功事象が互いに独立の場合には，n 個の二項分布の集まりなので，ベイズ推定に基づけば，タスク成功確率は：

$$\hat{p}_i = \frac{m_i + 1}{m + 2}$$

により推定されることになる．そして，このときにも上記と同じ議論が成り立つ．

5. 実験による評価

最尤推定法とベイズ推定法を比較するため，10 個のタスクから成る問題を作って実験を行った．タスクの成功事象が互いに排反の場合には，ランダムな成功確率 (和は 1) と 0 から 100 までのランダムなコストを与えた (**表 1**)．タスクの成功事象が互いに独立の場合には，0.5 以下のランダムな成功確率と表 1 と同じコストを与えた (**表 2**)．ここで，後者の確率を 0.5 以下に制限したのは，成功確率の高いタスクが先頭に存在し，容易に最適に近いタスク列が生成されるのを避けるためである．実験では，4 種の方法を比較した．すなわち，ランダムなタスク列を用いる方法，ランダムに初期タスク列を設定した最尤推定法，等確率を仮定してコストの小さい順に初期タスク列を設定した最尤推定法，および，ベイズ推定法である．真の成功確率（表 1 ま

表 1 成功確率とコスト（排反ケース）
Table 1 Probabilities and costs (exclusive case).

タスク	1	2	3	4	5	6	7	8	9	10
成功確率 p_i	.15	.22	.14	.4	.3	.17	.12	.3	.3	.7
コスト c_i	32	95	24	6	80	26	91	5	55	19
$p_i/c_i \times 10^4$	46	23	60	71	4	66	13	57	5	38

表 2 成功確率とコスト（独立ケース）
Table 2 Probabilities and costs (independent case).

タスク	1	2	3	4	5	6	7	8	9	10
成功確率 p_i	.12	.42	.11	.18	.14	.23	.49	.12	.12	.31
コスト c_i	32	95	24	6	80	26	91	5	55	19
$p_i/c_i \times 10^4$	38	44	46	300	17	88	54	240	22	163

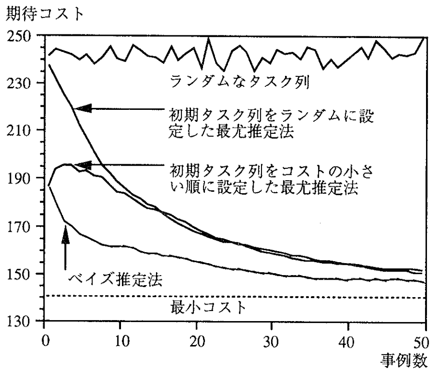


図 1 学習曲線（排反ケース）
Fig. 1 Learning curves (exclusive case).

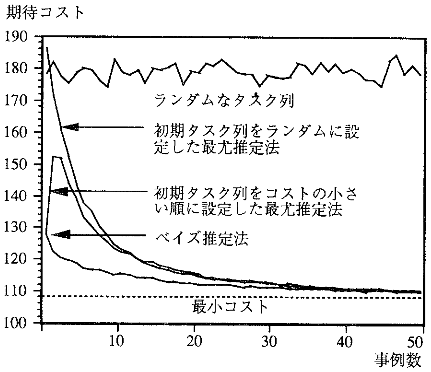


図 2 学習曲線（独立ケース）
Fig. 2 Learning curves (independent case).

たは2の値)に基づいてランダムに50個までの事例を生成して，それぞれの方法によりタスク列を逐次更新し，期待コストを計算した．

図1に，タスクの成功事象が互いに排反のケース，**図2**に，タスクの成功事象が互いに独立のケースの学習曲線を示す．ただし，結果は，試行を100回繰り返した平均値である．実験結果より，ベイズ推定法を用いれば，ランダムなタスク列や最尤推定法と比較して，任意の事例数の段階において，期待コストが最も小さいタスク列を生成できた．また，最小コストにも速く近付くことが分かった．最尤推定法では，初期タスク列をコストの小さい順に設定しても，初期タスク列をランダムに設定したときの曲線へ急速に近付くことが分かった．つまり，事例が少ない段階では，期待コストを改悪するのである．その理由は，既に指摘したように，事例が少ない段階では，最尤推定値の信頼性は低く，望ましくないタスクのペアの入れ換えが起こるためと考えられる．

6. タスク順序付け問題の一般化と事例実験

6.1 問題と解法の一般化

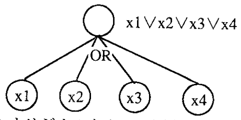
タスク順序付け問題の主たる応用は，μ-式（μ-formulae）[6]に属す論理式の真偽値を高速に判定する問題として一般化できる．ここで，μ-式とは，ブール木（Boolean trees）とも呼ばれ，各原子式（atomic fomula）が式に高々1回しか現れない論理式のクラスである．μ-式には，任意の論理積/和結合が許されるので，多くの自然な論理式をこのクラスで表現することができる．なお，これまでに述べたオリジナルなタスク順序づけ問題は，任意の原子式が論理和結合する特殊な場合に対応する．**図3**に，オリジナルと一般化したタスク順序づけ問題のAND/OR木の例を示す．また，木がμ-式で表現できるので，ゲームのMINIMAX探索への応用も可能である[5]．

一般化した問題の解法を説明する．ここで，原子式x_iの真偽値を判定するためのコストをc_i, x_iが真となる確率をp_iとする．ここでは，各原子式が真となる事象は互いに独立であると仮定する．まず，$f=x_1\vee\cdots\vee x_n$ならば，少なくとも1つのx_iが真となる成功確率p_f，および，原子式列$\sigma(i)$の順番で式を評価したときの判定に要する期待コストc_fは：

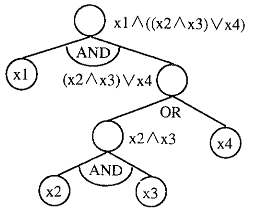
$$p_f=1-\prod_{i=1}^{n}(1-p_i)$$

$$c_f=c_{\sigma(1)}+\sum_{i=2}^{n}\left(\prod_{j=1}^{i-1}(1-p_{\sigma(j)})\right)c_{\sigma(i)}$$

となる．一方，$f=x_1\wedge\cdots\wedge x_n$ならば，すべての$x_i$が



(a) オリジナルなタスク順序づけ問題



(b) 一般化したタスク順序づけ問題

図3 タスク順序付け問題の一般化
Fig. 3 A general case of task sequencing problem.

真となる成功確率p_f，および，その判定に要する期待コストc_fは：

$$p_f=\prod_{i=1}^{n}p_i$$

$$c_f=c_{\sigma(1)}+\sum_{i=2}^{n}\left(\prod_{j=1}^{i-1}p_{\sigma(j)}\right)c_{\sigma(i)}$$

となる．従って，任意のμ-式に対して，葉から根に向けてこれらを再帰的に適用すれば，成功確率とコストが順番に定義できるので，μ-式の真偽値を高速に判定する問題は，タスク順序付け問題により解決することができる．なお，論理積結合した式については，速い段階で偽となる式を見つければ，後の処理を実行する必要がなくなるので，論理和結合の場合とは逆に，タスクを確率とコストの比の小さい順に並べれば，期待コストが最小になる．

6.2 事例実験

以下に，実験で用いるμ-式fを示す．

$$f=((x_1\vee x_2\vee x_3)\wedge(x_4\vee x_5\vee x_6\vee x_7))\vee((x_8\vee x_9\vee x_{10}\vee x_{11})\wedge(x_{12}\vee x_{13}\vee x_{14}))$$

ここで，fの真偽値を判定する問題は，7個のタスク順序付け問題に分解できる．すなわち，まず，原子式が論理和結合した4つの式の評価，次のレベルで論理積結合した2つの式の評価，および，トップレベルで論理和結合した式の評価で，合計7個である．実験では，3種の方法を比較した．すなわち，ランダムなタスク列に基づく方法，最尤推定法，および，ベイズ推定法である．なお，実験の枠組は，前回と同じ設定を用いた．**表3**に，実験で用いた成功確率とコスト，**図4**に，結果の学習曲線を示す．ベイズ推定法を用いれば，ランダムなタスク列や最尤推定法と比較して，任意の事例数の段階において，最も期待コストが小さくなった．また，事例が増えても最尤推定法との差は明らかであ

表 3 原子式の確率とコスト
Table 3 Probabilities and costs of atomic formulae.

原子式	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}
確率 p_i	.68	.96	.35	.18	.73	.34	.79	.66	.14	.54	.75	.85	.22	.04
コスト c_i	20	41	41	22	47	34	90	87	24	41	75	49	86	3

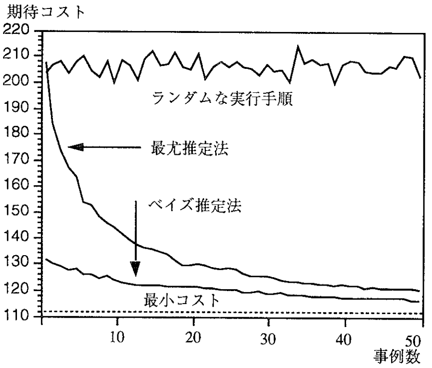


図 4 論理式に対する学習曲線
Fig. 4 Learning curves for a formula.

り，その理由は，複数のタスク列最適化問題の効果が相乗しているためと考えられる．従って，問題の規模が大きく複雑になれば，ベイズ推定法を用いるメリットは，さらに大きくなると期待される．

7. おわりに

本論文では，タスク順序付け問題，および，この問題を一般化した論理式の真偽値判定問題において，タスク成功確率をベイズ推定で求める方法が最尤推定を用いる方法より優ることを示した．今後は，より一般的な枠組において，確率推定法の理論的な解析，および，有効なアルゴリズムの検討を行う予定である．

参 考 文 献

1) Cestnik, B.: Estimating Probabilities: A Crucial Task in Machine Learning, *Proceedings of the European Conference on Artificial Intelligence*, pp. 147–149, Stockholm, Sweden (1990).
2) Duda, R. and Hart, H.: *Pattern Classification and Scene Analysis*, John Wiley & Sons, (1973).
3) Garey, M.: Optimal Task Sequencing with Precedence Constraints, *Discrete Math.*, Vol. 4, pp. 37–56 (1973).
4) Good, I.: *The Estimation of Probabilities: An Essay on Modern Bayesian Methods, Research Monograph 30*, MIT Press (1965).
5) Pearl, J.: *Heuristics*, Addison-Wesley (1984).
6) Pitt, L. and Valiant, L.: Computational Limitations on Learning from Examples, *J. ACM*, Vol. 35, No. 4, pp. 965–984 (1988).
7) Simon, H. and Kadane, J.: Optimal Problem-Solving Search: All-or-none Solution, *Artif. Intell.*, Vol. 6, No. 3, pp. 235–247 (1975).
8) Slagle, J.: An Efficient Algorithm for Finding Certain Minimum-cost Procedure for Making Binary Decisions, *J. Assoc. Comput. Machinery*, Vol. 11, No. 3, pp. 253-264 (1964).

付　　録

定理 2 の証明を示す．

証明 タスク成功確率を $\bar{p}=(p_1, \cdots, p_n)$，$m$ 個の事例が現れた後，各タスクが成功した事例数を $\bar{m}=(m_1, \cdots, m_n)$ とする．まず，タスクの成功事象が互いに排反の場合を考える．ベイズ推定では，タスクが $\bar{m}$ 回成功したという条件の下で，タスク成功確率が $\bar{p}$ となる確率密度関数は：

$$f(\bar{p}|\bar{m})=\frac{(m+n-1)!}{\sqrt{n}\,m_1!\cdots m_n!}p_1^{m_1}\cdots p_n^{m_n}$$

となる[2]．ここで，タスク T_i, T_j を (i, j) という順序にすれば，最小コストとの差を表す確率変数 $X_{(i,j)}$ は：

$$X_{(i,j)}=\begin{cases} c_ip_j-c_jp_i & c_ip_j>c_jp_i \\ 0 & c_ip_j\leq c_jp_i \end{cases}$$

となり，逆の順序 (j, i) では，確率変数 $X_{(j,i)}$ は：

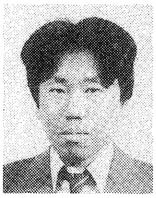
$$X_{(j,i)}=\begin{cases} 0 & c_ip_j\geq c_jp_i \\ c_jp_i-c_ip_j & c_ip_j<c_jp_i \end{cases}$$

となる．ここで，それぞれの $f(\bar{p}|\bar{m})$ に対する平均値の差を求めれば，

$$\begin{aligned} &E(X_{(j,i)})-E(X_{(i,j)}) \\ &\quad=\int_{p_1+\cdots+p_n=1}(c_jp_i-c_ip_j)f(\bar{p}|\bar{m})dp_1\cdots dp_n \\ &\quad=\frac{c_j(m_i+1)-c_i(m_j+1)}{m+n} \end{aligned}$$

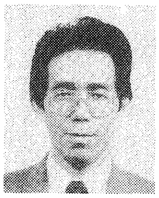
となる．従って，タスクを $(m_i+1)/c_i$ の大きい順に並べれば，平均期待コストが最小になる． □

(平成 6 年 1 月 31 日受付)
(平成 7 年 1 月 12 日採録)



斉藤　和巳（正会員）

1963年生．1985年慶應義塾大学理工学部数理科学科卒業．同年，NTT入社．1991年より1年間オタワ大学客員研究員．神経回路網，機械学習の研究に従事．現在，NTTコミュニケーション科学研究所主任研究員．人工知能学会，神経回路学会，日本認知科学会各会員．



中野　良平（正会員）

1947年生．1971年東京大学工学部計数工学科卒業．工学博士．同年，日本電信電話公社（現NTT）入社．以来，統計解析，分散処理，データベース，人工知能の研究に従事．現在，NTTコミュニケーション科学研究所主幹研究員．人工知能学会，神経回路学会，日本応用数理学会各会員．