

# Web 情報を活用した自動献立作成のための 多目的遺伝的アルゴリズムによる 並列分散処理

Parallel Distributed Processing  
of a Multiobjective Genetic Algorithm  
for Automatic Menu Planning Using Web Information

安藤 祐斗 (Yuto Ando)  
t815008@st.pu-toyama.ac.jp

富山県立大学 工学部 電子・情報工学科 情報基盤工学講座

Teams, 14:45-15:00 Wednesday, February 16, 2022.

1. はじめに
2. 自動献立作成の概要
3. 制約条件を考慮できる多目的遺伝的アルゴリズム
4. 提案手法
5. 数値実験並びに考察
6. おわりに

## 1.1 本研究の背景

2/16

1. はじめに
2. 自動献立作成の概要
3. 制約条件を考慮できる多目的遺伝的アルゴリズム
4. 提案手法
5. 数値実験並びに考察
6. おわりに

近年、生活習慣病を患う人々が増加している。生活習慣病とは「食習慣、運動習慣、休養、喫煙、飲酒、ストレスなどの生活習慣を原因として発症する疾患の総称」のことであり、深刻な疾患に深く関与している。これを防ぐためには、過度な喫煙や飲酒を控えることや、しっかりと栄養を食事から摂取することが重要視されている。また、学校給食や病院食の現場で献立作成業務を担当している栄養士は、摂るべき栄養素の計算や、食事にかかる金額などを設定する作業を、繰り返し見直す必要があるため、献立作成の負荷は高いことがわかる。



図1 生活習慣病を起因とする疾患

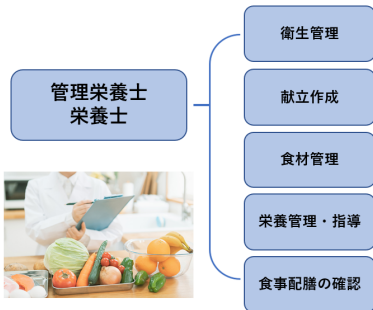


図2 栄養士の主な業務内容



## 1.2 本研究の目的

3/16

そこで、人によって摂るべき栄養素やカロリーが満たされた 1 週間分の献立作成をコンピュータによって自動的に行い、さらに並列分散処理を施すことによって実行速度の向上を図るシステムを提案する。

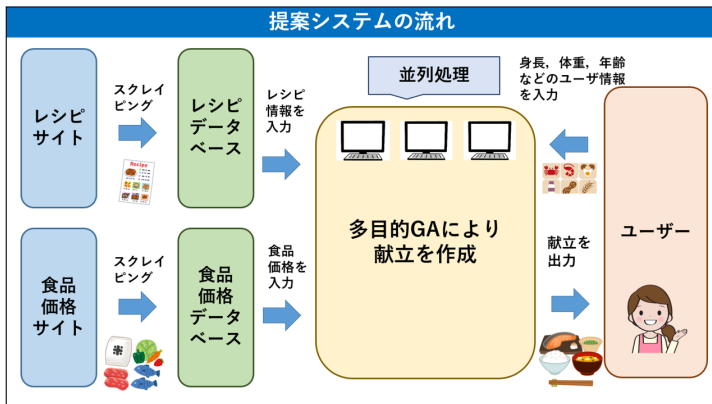


図3 本研究における提案システムの流れ

1. はじめに
2. 自動献立作成の概要
3. 制約条件を考慮できる多目的遺伝的アルゴリズム
4. 提案手法
5. 数値実験並びに考察
6. おわりに

## 2.1 自動献立作成支援システムの概要

4/16

自動献立作成支援システムとは、献立を自動で生成することができるシステムである。献立を構成する要素として主菜と副菜を用意する。また、時間帯別の調理時間アンケートを数値実験の参考にする。

- はじめに
- 自動献立作成の概要
- 制約条件を考慮できる多目的遺伝的アルゴリズム
- 提案手法
- 数値実験並びに考察
- おわりに



図4 献立作成アプリケーションの例



図5 主菜と副菜のイメージ

表1 時間帯別の調理時間アンケート

	n	15分未満	15～30分	31分～60分	61分以上	無回答
朝食	10043	70%	24.2%	4.1%	0.4%	1.3%
昼食	10074	47%	37.0%	14.7%	0.8%	0.5%
夕食	10151	8.7%	47.3%	36.6%	5.9%	0.9%

※nはアンケート回答者数を指す

## 2.2 Web 上のレシピデータを活用

5/16

1. はじめに
2. 自動献立作成の概要
3. 制約条件を考慮できる多目的遺伝的アルゴリズム
4. 提案手法
5. 数値実験並びに考察
6. おわりに

レシピサイト「ポプとアンジー」から、主菜と副菜カテゴリに含まれる料理と、朝食でキーワード検索した料理レシピデータ（必要材料、摂取栄養量、カロリーなど）をスクレイピングし、食品価格動向を調査しているサイト「小売物価統計調査による価格調査」から様々な食品とその価格データをスクレイピングする。次に、料理レシピデータの食材と食材価格データの食材を照らし合わせて食材コストを計算する。



図7 レシピサイト・ポプとアンジーにおける料理レシピ情報

### スクレイピングする主なデータ

- ・料理レシピ名
- ・調理時間
- ・摂取カロリー
- ・摂取栄養量
- ・必要食材名
- ・必要食材量
- ・作り方
- ・画像URL
- ・食材価格
- ・販売単位
- ・食材名

小売物価統計調査による価格推移

HOME / キャベツの販売価格・価格推移 / 全国

全国のキャベツ 1kg  
価格推移 / 過去84ヵ月



WEBスクレイピング

データ抽出・出力

全国のスーパーで売られているキャベツ 1kg の価格の平均は128円。  
2015年1月～2021年12月(過去84ヵ月)の期間で全国のキャベツが最も高かった最高価格は2018年2月で354円。逆に最もキャベツが安かった最安値月は2023年2月で118円となっています。  
全国よりキャベツの最高値(2018年2月)と最安値(2023年2月)との価格差は276.4321円となっています。  
キャベツ 1kg の2015年1月～2021年12月の価格推移とグラフは下記をご覧ください。  
出典元：総務省統計局 小売物価統計調査(2021年12月)  
関連新聞：キャベツのふくらみと価格(ヨミランタン)

図8 食品価格推移調査サイトの例



食材コスト  
の計算

図9 Webデータ活用の流れ

## 2.3 多目的最適化による自動献立作成

6/16

献立作成システムは、決められた制約条件の中で、目的関数を最大または最小となるパラメータの、組み合わせの解を探索する、組み合わせ最適化問題として捉えられる。献立作成における制約条件として、何日分の献立を作成するか、カロリーをどのくらい制限するか、などが挙げられる。また、目的関数として、調理時間の最小化や個人の嗜好の最大化などが挙げられる。

### 献立作成における 目的関数、制約条件の例

#### 目的関数の例

- ・ 調理時間の最小化
- ・ 個人の嗜好の最大化
- ・ 食材ロスの最小化
- ・ 食材コストの最小化

#### 制約条件の例

- ・ 特定の栄養素量の制限
- ・ 摂取カロリーの制限
- ・ 献立を作成する日数
- ・ 調理工程の制限

### 献立作成における研究例

- ・ 必要な摂取栄養量をファジィ数で表す、ファジィ数理計画法を用いた献立作成。
- ・ ユーザとの対話型処理によって献立を作成する。
- ・ ユーザの献立の雰囲気によって献立を作成する。



図10 ナップサック問題の例

### カレー作りのPERT図



図11 PERT図の例

1. はじめに
2. 自動献立作成の概要
3. 制約条件を考慮できる多目的遺伝的アルゴリズム
4. 提案手法
5. 数値実験並びに考察
6. おわりに

### 3.1 多目的最適化とパレート最適解

7/16

多目的最適化は、ある制約条件のもと、複数の目的関数を最大化、あるいは最小化する手法である。全ての目的関数を最大化、あるいは最小化するような最適解が存在するとは言えないため、パレート最適という概念を導入する必要がある。

#### 多目的最適化の定式化

minimize  
 $x$

$\{f_1(x), f_2(x), \dots, f_n(x)\}$

subject to

$g_k(x) \leq 0$

$k = 1, 2, \dots, m$

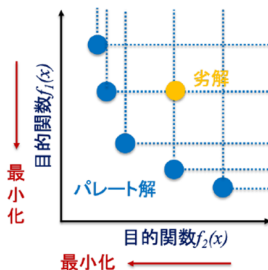


図12 パレート解のイメージ

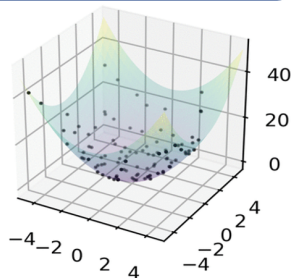


図13 解探索のイメージ (粒子群最適化)

1. はじめに
2. 自動献立作成の概要
3. 制約条件を考慮できる多目的遺伝的アルゴリズム
4. 提案手法
5. 数値実験並びに考察
6. おわりに

## 3.2 遺伝的アルゴリズムによる多目的最適化

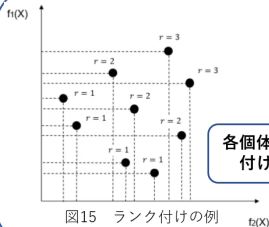
8/16

多目的最適化問題を解く手法として、NSGA-IIを用いる。これは、遺伝的アルゴリズムを多目的最適化問題に拡張したものであり、非優越ソート、混雑度トーナメント選択といった特徴を持つ。

### NSGA-IIの特徴

- ・非優越ソート
- ・混雑度トーナメント選択

### 非優越ソート



各個体にランク  
付けを行う

### 混雑度トーナメント選択

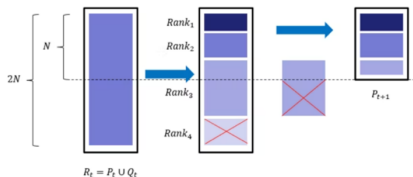


図14 NSGA-IIのアルゴリズム

### 混雑度トーナメント選択

$$\text{混雑距離} : CD(x^{(i)}) = \frac{1}{k} \sum_{j=1}^k |\tilde{f}_j(x^{i+1}) - \tilde{f}_j(x^{i-1})|$$

- ・個体  $i$  のランクが個体  $j$  のランクよりも優れている。
- ・個体  $i$  と個体  $j$  はともに同じランクであり、 $i$  の混雑距離が  $j$  よりも優れている。

1. はじめに
2. 自動試立作成の概要
3. 制約条件を考慮できる多目的遺伝的アルゴリズム
4. 提案手法
5. 数値実験並びに考察
6. おわりに



### 3.3 並列分散処理による解法

9/16

1. はじめに
2. 自動献立作成の概要
3. 制約条件を考慮できる多目的遺伝的アルゴリズム
4. 提案手法
5. 数値実験並びに考察
6. おわりに

最適化問題において大量のデータを扱う場合、処理速度が低下してしまうため、処理速度の向上に努める必要がある。並列分散処理とは、複数台のコンピュータを用いて1つの計算処理を行い、性能や処理速度の向上させることで、並列分散ソフトウェアである、Daskを用いて献立作成における最適化部分の並列分散処理を行う。

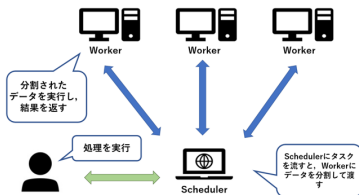


図16 並列分散処理のイメージ

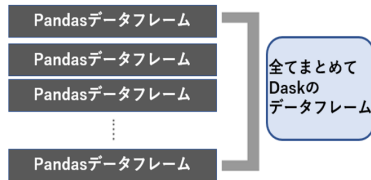


図18 Daskデータフレーム

Numpyがグリッド上に配置されたものが1つのDask配列となる



図19 Dask配列



図17 並列分散ソフトウェア「Dask」

## 4.1 調理時間とコストを最小化するパレート最適な献立

10/16

献立に含まれる料理の調理時間の最小化と、料理の食材コストの最小化を目的関数とし、摂取カロリーや摂取栄養量などを制約条件として多目的最適化を行い、パレート最適な献立を出力する。

### 定式化

$$\begin{aligned}
 &\text{minimize} && \sum_{k=1}^{3D} \sum_{i=1}^R r_{ki} T_i \\
 &\text{minimize} && \sum_{k=1}^{3D} \sum_{i=1}^R r_{ki} G_i \\
 &\text{subject to} && F_i^L \leq \sum_i r_{ki} f_{il} \leq F_i^H \quad (\forall k, \forall l) \\
 & && B^L \leq \sum_i r_{ki} C_i \leq B^H \quad (\forall k) \\
 & && \sum_i r_{ki} T_i \leq \tau_1 \quad (k \% 3 = 1) \\
 & && \sum_i r_{ki} T_i \leq \tau_2 \quad (k \% 3 = 2) \\
 & && \sum_i r_{ki} T_i \leq \tau_3 \quad (k \% 3 = 3) \\
 & && 0 < \sum_i r_{ki} \sigma_i \leq 1 \quad (\forall k) \\
 & && 0 \leq \sum_i r_{ki} (1 - \sigma_i) \leq 3 \quad (\forall k) \\
 & && \sum_{k=1}^{3D} r_{ki} \leq 1
 \end{aligned}$$

調理時間の最小化

食材コストの最小化

摂取栄養量の制約

摂取カロリーの制約

調理時間の制約(朝)

調理時間の制約(昼)

調理時間の制約(夜)

主菜の制約

副菜の制約

料理種類の制約

### 変数

対象の日数:  $D$   
 レシピの数:  $R$   
 食材の数:  $Q$   
 栄養素の数:  $N$   
 データベース上の食材数:  $S$   
 データベース上の食材番号:  $d: 1, 2, 3, \dots, S$   
 日の番号:  $k: 1, 2, 3, \dots, 3D$   
 栄養素の番号:  $l: 1, 2, 3, \dots, N$   
 材料の番号:  $m: 1, 2, 3, \dots, Q$   
 レシピの番号:  $i: 1, 2, 3, \dots, R$   
 $i$  番目のレシピの名前:  $y_i$   
 $i$  番目のレシピの献立フラグ:  $r_{ki}$   
 $i$  番目のレシピの主菜フラグ:  $\sigma_i$   
 $i$  番目のレシピの調理時間:  $T_i$   
 $i$  番目のレシピの摂取カロリー:  $C_i$

### ログイン情報

使用者の名前:  $\epsilon$   
 身長:  $\alpha$   
 体重:  $\beta$   
 基礎代謝量(下限):  $B^L$   
 基礎代謝量(上限):  $B^H$

$i$  番目のレシピの食材コスト:  $G_i = \sum_{m=0}^Q \sum_{d=q_{im}} \frac{e_{im}}{W_d} M_d$   
 $i$  番目のレシピの  $m$  番目の材料の名前:  $q_{im}$   
 $i$  番目のレシピの  $m$  番目の材料量:  $e_{im}$   
 $i$  番目のレシピの  $l$  番目の栄養素の名前:  $n_{il}$   
 $i$  番目のレシピの  $l$  番目の栄養素の量:  $f_{il}$   
 $d$  番目の食材名:  $Z_d$   
 $d$  番目の食材の販売単位:  $W_d$   
 $d$  番目の食材の値段:  $M_d$

図20 献立作成における多目的最適化の定式化

## 4.2 対話による最適な献立の提示

11/16

多目的最適化によって得られた、調理時間と材料コストが最小化されたパレート最適解（レシピ）の中から、利用者が選択して献立を作成する際に対話型処理を行う。Tkinter を用いて、0 に近づくほど調理時間の最小化を重視した献立を、1 に近づくほど食材コストを重視して最小化した 1 週間分の献立を出力されるように設定するためのスライダーを作成する。



図21 対話型処理のイメージ

### 正規化

データの最小値を0, 最大値を1にスケールリングする手法。

$i = 1 \dots n$ ,  $i$  番目の観測値を  $x_i$ , 全観測値を  $x$ , 正規化した値を  $x'$  とすると以下の式で与えられる。

$$x'_i = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

パレート解に含まれる,  
1週間分の調理時間合計  
と食材コスト合計を  
正規化

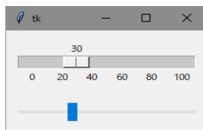


図22 スライダーの例

調理時間と食材コストの  
どちらかを重視する方向  
にスライダーを動かす

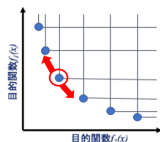


図23 パレート解

パレート解からスライダー  
に則った献立が出力される

## 4.3 提案システムの構成

12/16

### 動画

提案システムのアルゴリズムを実装した開発システムの概要を動画で示す.

1. はじめに
2. 自動献立作成の概要
3. 制約条件を考慮できる多目的遺伝的アルゴリズム
4. 提案手法
5. 数値実験並びに考察
6. おわりに

## 5.1 数値実験の概要

13/16

データベースに蓄積した，料理レシピ，食材の価格データの例を示す．また，実際に作成した 1 週間分の献立の，食材コストの合計と調理時間の合計を縦軸と横軸に設定したパレート解と，対話型処理に用いたスライダーを示す．

レシピの名前	主要フラグ	調理時間	コスト	摂取カロリー	材料名	材料量	栄養素名	栄養量
ポークソテーりんごソース	1	15	306	236	豚ロース肉厚切り(脂身なし)	120g	たんぱく質	12.3g
NAN	NAN	NAN	NAN	NAN	塩	少々	炭水化物	10g
NAN	NAN	NAN	NAN	NAN	こしょう	少々	糖質	9g
NAN	NAN	NAN	NAN	NAN	小麦粉	小さじ2	脂質	15.6g
NAN	NAN	NAN	NAN	NAN	油	小さじ2	食物繊維	0.9g
NAN	NAN	NAN	NAN	NAN	りんご	60g	ビタミンA	17μg
NAN	NAN	NAN	NAN	NAN	トマトケチャップ	小さじ4	ビタミンB1	0.44mg
NAN	NAN	NAN	NAN	NAN	さやいんげん	30g	ビタミンB2	0.11mg
NAN	NAN	NAN	NAN	NAN	NAN	NAN	ビタミンB6	0.22mg
NAN	NAN	NAN	NAN	NAN	NAN	NAN	ビタミンB12	0.2μg
NAN	NAN	NAN	NAN	NAN	NAN	NAN	ビタミンC	4mg
NAN	NAN	NAN	NAN	NAN	NAN	NAN	ビタミンD	0.1μg
NAN	NAN	NAN	NAN	NAN	NAN	NAN	ビタミンE	1mg
NAN	NAN	NAN	NAN	NAN	NAN	NAN	ビタミンK	18μg
NAN	NAN	NAN	NAN	NAN	NAN	NAN	カリウム	306mg
NAN	NAN	NAN	NAN	NAN	NAN	NAN	カルシウム	13mg
NAN	NAN	NAN	NAN	NAN	NAN	NAN	鉄	0.4mg

食材名	値段	販売単位
キャベツ	171円	1 k g
ほうれん草	985円	1 k g
白菜	245円	1 k g
ねぎ	703円	1 k g
レタス	450円	1 k g
もやし	173円	1 k g
ブロッコリー	675円	1 k g
アスパラガス	1777円	1 k g
さつまいも	586円	1 k g
じゃがいも	406円	1 k g

図24 レシピデータの例

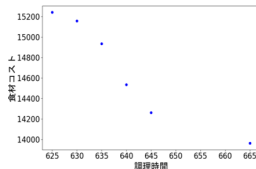


図26 出力されたパレート解

ウインドウ

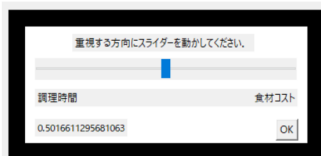


図27 作成したスライダー

- はじめに
- 自動献立作成の概要
- 制約条件を考慮できる多目的遺伝的アルゴリズム
- 提案手法
- 数値実験並びに考察
- おわりに

## 5.2 実験結果と考察 (制約条件, 献立のパラメータ)

14/16

自分の身長を 160cm, 体重を 58kg, 年齢を 23 歳, 性別を男, 身体活動レベルを普通として入力とし, 1 日に必要な推定エネルギー量を算出し, 目的関数と制約条件に基いて 1 週間分の献立作成を行った結果を示す。

< 基礎代謝量 >

$$\text{基礎代謝量 (kcal/日)} = \text{基礎代謝基準値} \times \text{体重 (kg)}$$

< 必要推定エネルギー量 >

$$\text{必要推定エネルギー量 (kcal/日)} = \text{基礎代謝量} \times \text{身体活動レベル指数}$$

< 必要たんぱく質 >

$$\text{たんぱく質 (g/日)} = \frac{\text{必要推定エネルギー量 (kcal/日)} \times 0.13}{4(g)}$$

< 必要脂質 >

$$\text{脂質 (g/日)} = \frac{\text{必要推定エネルギー量 (kcal/日)} \times 0.15}{9(g)}$$

< 必要炭水化物 >

$$\text{炭水化物 (g/日)} = \frac{\text{必要推定エネルギー量 (kcal/日)} \times 0.4}{4(g)}$$

厚生労働省「日本人の食事摂取基準」  
2020年度版より

性別	男性		
	基礎代謝 基準値 (kcal/kg体重/日)	参照体重 (kg)	基礎 代謝量 (kcal/日)
年齢			
18-29	24.0	63.2	1520
30-49	22.3	68.5	1530
50-69	21.5	65.3	1400
70 以上	21.5	60.0	1290

表2 設定された制約条件

制約条件	設定した値
摂取カロリー (kcal)	2336~2536
たんぱく質 (g)	75.92
脂質 (g)	38.93
炭水化物 (g)	233.6
調理時間合計 [朝] (分)	15 以下
調理時間合計 [昼] (分)	45 以下
調理時間合計 [夕] (分)	60 以下
主菜	1(昼, 夕)
副菜	2(昼, 夕)

表3 1週間分の献立のパラメータ

	1日目	2日目	3日目	4日目	5日目	6日目	7日目
摂取カロリー (kcal)	2461	2345	2491	2491	2397	2394	2352
たんぱく質 (g)	88.9	80.5	97.5	100.7	77.9	107.5	87
脂質 (g)	88.4	58.5	56.6	89.2	89.2	74.5	56.9
炭水化物 (g)	278.9	332.6	366	284	279	299	360.3
調理時間合計 [朝] (分)	10	10	10	10	15	15	10
調理時間合計 [昼] (分)	40	40	40	45	30	35	40
調理時間合計 [夕] (分)	35	35	45	35	55	35	50

・表2と表3を比較すると,  
制約条件を満たしながら  
1週間分の献立作成が  
できていることが分かる。

- はじめに
- 自動献立作成の概要
- 制約条件を考慮できる多目的遺伝的アルゴリズム
- 提案手法
- 数値実験並びに考察
- おわりに

## 5.2 実験結果と考察 (献立イメージ, 並列分散処理の結果)

15/16

作成した 1 週間分の献立を html 上に出力した結果を示す。また, 1 台, 2 台, 4 台の PC で並列分散処理を行った処理時間の結果を示す。



図28 htmlに出力された献立の例(1日目)

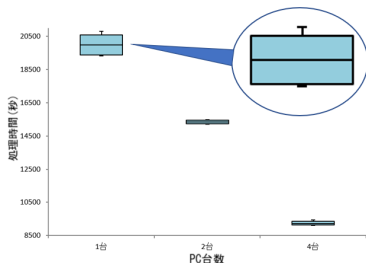


図29 PC台数別における処理時間の箱ひげ図

表4 並列分散処理の結果

台数	1回目	2回目	3回目	4回目	5回目	平均	標準偏差
1	5h24m	5h22m	5h46m	5h33m	5h39m	5h33m	9m18s
2	4h14m	4h13m	4h16m	4h17m	4h15m	4h15m	1m38s
4	2h32m	2h33m	2h32m	2h34m	2h37m	2h34m	1m47s

- はじめに
- 自動献立作成の概要
- 制約条件を考慮できる多目的遺伝的アルゴリズム
- 提案手法
- 数値実験並びに考察
- おわりに

## おわりに

**本研究では、Web上のデータを活用し、1週間分の献立を最適化によって作成したのちに、対話型処理によってその中から選択できるシステムを提案した。**

- 入力したユーザ情報によって設定された制約条件を満たしながら、2つの目的関数について最小化された1週間分の献立を作成した。
- スライダーを動かすことによって、対話型で1週間分の献立をパレート解から選択できるようなシステムを構成した。
- 自動献立作成システムの最適化処理部分について、並列分散処理を行なうことで、処理速度を向上させた。

## 今後の課題

- 献立のバリエーションを増やすため、他の料理レシピサイトからスクレイピングしたレシピデータも扱えるようにする。
- 4台で並列分散処理を行った場合でも、処理時間に2時間30分はかかってしまうため、並列分散処理以外の手法を扱うなどの、処理速度を速くするための工夫を加える。