

(19) 日本国特許庁 (JP)

(12) 特 許 公 報 (B2)

(11) 特許番号

特許第6251663号  
(P6251663)

(45) 発行日 平成29年12月20日 (2017.12.20)

(24) 登録日 平成29年12月1日 (2017.12.1)

(51) Int.Cl.

F I

G 0 6 Q 10/06 (2012.01)

G 0 6 Q 10/06 3 0 0

請求項の数 7 (全 19 頁)

(21) 出願番号	特願2014-224801 (P2014-224801)	(73) 特許権者	000004226
(22) 出願日	平成26年11月5日 (2014.11.5)		日本電信電話株式会社
(65) 公開番号	特開2016-91286 (P2016-91286A)		東京都千代田区大手町一丁目5番1号
(43) 公開日	平成28年5月23日 (2016.5.23)	(74) 代理人	100086232
審査請求日	平成28年12月8日 (2016.12.8)		弁理士 小林 博通
		(74) 代理人	100104938
			弁理士 鶴澤 英久
		(74) 代理人	100140361
			弁理士 山口 幸二
		(72) 発明者	羽田野 真由美
			東京都千代田区大手町一丁目5番1号 日
			本電信電話株式会社内
		(72) 発明者	中辻 真
			東京都千代田区大手町一丁目5番1号 日
			本電信電話株式会社内

最終頁に続く

(54) 【発明の名称】 タスク割当て装置、タスク割当て方法、タスク割当てプログラム

(57) 【特許請求の範囲】

【請求項1】

時刻IDとワーカIDとタスクIDとが対応付けられた第1データベースと、  
タスクが発生する位置情報が格納された第2データベースと、  
第1データベースからワーカIDとタスクIDのペアと、該ペアの時刻IDとを抽出する手段と、

抽出されたタスクIDをキーにして第2データベースからタスクの位置情報を検索する手段と、

抽出されたワーカIDをキーにしてワーカの時刻毎の位置情報が格納された第3データベースから時刻毎のワーカの位置情報を検索する手段と、

時刻IDが示す時刻毎にワーカがタスクの発生する位置に移動した場合の時刻を算出する手段と、

算出された時刻のうち最も早い時刻と、該時刻に対応するワーカとタスクと該時刻の時刻IDとを選択して出力する手段と、

を備えることを特徴とするタスク割当て装置。

【請求項2】

前記第3データベースには、さらにワーカID毎に移動手段が格納され、  
前記移動手段を用いて時刻IDの示す時間にワーカが、タスクの発生する位置に移動した場合の時刻を、

式(1)(2)により算出する手段を備えることを特徴とする請求項1記載のタスク割

10

20

当て装置。

【数 1】

$$duration_k = \frac{dist_k}{v_k}, \quad k = 1, 2, 3 \quad \text{式(1)}$$

$duration_k$  : ある時刻  $t_k$  にワーカがタスクの位置まで移動する所要時間 ,

$dist_k$  : 各時間におけるタスクの発生する位置までの距離 ,

$v_k$  : 移動速度

【数 2】

10

$$time_k = \tau_k + duration_k, \quad k = 1, 2, 3 \quad \text{式(2)}$$

【請求項 3】

前記第 3 データベースには、ワーカ ID 毎に最大移動距離が格納されているとともに、  
前記 2 データベースおよび前記第 3 データベースに基づきワーカの最大移動距離内にタ  
スクが存在するか否かを判定し、時刻 ID とワーカ ID とタスク ID とを対応付けて前記  
第 1 データベースに格納する手段と、

前記選択された最も早い時刻とワーカとタスクと時刻 ID とを対応付けて格納する第 4  
データベースと、

20

該第 4 データベースに基づきワーカ対タスクの最大割当て問題を 2 部グラフの最小コス  
トフロー問題に置き換えたときに与えられる制約式を作成し、作成された制約式を第 5 デ  
ータベースに格納させる手段と、

前記第 4 データベースおよび前記第 5 データベースに基づき時刻 ID 毎にタスク完了ま  
での時間が最小となる時刻・ワーカ・タスクのペアを算出する手段と、

をさらに備えることを特徴とする請求項 2 記載のタスク割当て装置。

【請求項 4】

コンピュータを用いて複数のタスクを複数のワーカに振り分けるタスク割当て方法であ  
って、

30

時刻 ID とワーカ ID とタスク ID とが対応付けられた第 1 データベースに基づきワー  
カ ID とタスク ID のペアと、該ペアの時刻 ID とを抽出する第 1 ステップと、

抽出されたタスク ID をキーにしてタスクが発生する位置情報が格納された第 2 データ  
ベースからタスクの位置情報を検索する第 2 ステップと、

抽出されたワーカ ID をキーにしてワーカの時刻毎の位置情報が格納された第 3 データ  
ベースから時刻毎のワーカの位置情報を検索する第 3 ステップと、

時刻 ID が示す時刻毎にワーカがタスクの発生する位置に移動した場合の時刻を算出す  
る第 4 ステップと、

算出された時刻のうち最も早い時刻と、該時刻に対応するワーカとタスクと該時刻の時  
刻 ID とを選択して出力する第 5 ステップと、

40

を有することを特徴とするタスク割当て方法。

【請求項 5】

前記第 3 データベースには、さらにワーカ ID 毎に移動手段が格納されている場合にお  
いて、

前記第 4 ステップは、前記移動手段を用いて時刻 ID の示す時刻にワーカがタスクの発  
生する位置に移動した場合の時刻を、

式(1)(2)により算出するステップを有することを特徴とする請求項 4 記載のタス  
ク割当て方法。

【数 1】

$$duration_k = \frac{dist_k}{v_k}, \quad k = 1, 2, 3 \quad \text{式(1)}$$

$duration_k$  : ある時刻  $k$  にワーカがタスクの位置まで移動する所要時間,

$dist_k$  : 各時間におけるタスクの発生する位置までの距離,

$v_k$  : 移動速度

【数 2】

$$time_k = \tau_k + duration_k, \quad k = 1, 2, 3 \quad \text{式(2)}$$

【請求項 6】

前記第 3 データベースにはワーカ ID 毎に最大移動距離が格納され、

前記選択された最も早い時刻とワーカとタスクと時刻 ID とが対応付けられて第 4 データベースに格納されている場合において、

前記第 2 データベースおよび前記第 3 データベースに基づきワーカの最大移動距離内にタスクが存在するか否かを判定し、時刻 ID とワーカ ID とタスク ID とを対応付けて前記第 1 データベースに格納するステップと、

前記第 4 データベースに基づきワーカ対タスクの最大割当て問題を 2 部グラフの最小コストフロー問題に置き換えたときに与えられる制約式を作成し、作成された制約式を第 5 データベースに格納させるステップと、

前記第 4 データベースおよび前記第 5 データベースに基づき時刻 ID 毎にタスク完了までの時間が最小となる時刻・ワーカ・タスクのペアを算出するステップと、

をさらに有することを特徴とする請求項 5 記載のタスク割当て方法。

【請求項 7】

請求項 1 ~ 3 記載のタスク割当て装置として、コンピュータを機能させることを特徴とするタスク割当てプログラム。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、コンピュータを用いて複数の仕事を複数人に振り分けるタスク割当て処理の技術に関する。

【背景技術】

【0002】

現在、複数の仕事（以下、タスクという。）を複数の人（以下、ワーカという。）に割り振る際に 1 人あたりのタスク数の上下限や、1 タスクあたりのワーカ数の上下限などの制約条件を満たしながらワーカ対タスクの割当て数を最大にする最適化計算を行うことで効率的な割当てを実現する取り組みが行われている。

【0003】

さらに、あるワーカ「 $i$ 」が、あるタスク「 $j$ 」を受け持つときにコスト「 $c_{ij}$ 」が必要な場合、総コストが最小となる最適化計算を行うことで効率的な割当てを実現する取り組みが行われている。これらのワーカ対タスクの最適な割当て方法を見つける問題は、「割当て問題」と呼ばれる。

【0004】

一般に、ワーカ対タスクの「最大割当て問題」は、2 部グラフ（頂点集合が 2 分割され、辺がその部分集合の間を結ぶものに限られるグラフ）の 2 つの頂点集合にそれぞれワーカとタスクを対応させ、割当て可能なワーカとタスクの頂点間の辺を結ぶことで、「2 部グラフの最大マッチング問題」に置き換えることができる。この「2 部グラフの最大マッ

10

20

30

40

50

チング問題」とは、与えられた制約条件のもと2部グラフの割当て数を最大にするペアを選択するものである。

【0005】

また、「2部グラフの最大マッチング問題」は、図1に示すように、ソース(src)となる頂点を新たに加え、ワーカが対応した頂点集合に向かう辺を追加するとともに、シンク(sink)となる頂点を新たに加え、タスクが対応した頂点集合からの辺を追加することで、ソースからシンクへ流れる流量を最大化する「最大フロー問題」に変換することが可能となる。なお、ワーカ対タスクの「最大割当て最小コスト問題」は、先ほどと同様にグラフを変換してソースからシンクへ流れる流量を最大化したときに、コストが最小となる辺を選ぶ「最小コストフロー問題」に変換することができる。

10

【0006】

「最大フロー問題」、「最小コストフロー問題」のいずれも線形計画法で解くことができ、特に「最大フロー問題」を解くための効率的なアルゴリズムが非特許文献1に複数提案されている。

【0007】

以下、クラウドソーシングサービスにおけるワーカ対タスクの割当て問題を解く際の背景技術を説明する。ここでクラウドソーシングとは、不特定多数のワーカにタスクを委託する枠組みのことであり、例えば非特許文献2のランサーズのように既に商用サービスとして成立しているものもある。

20

【0008】

その中でも特に、タスクに時空間条件が付与されており、ワーカが指定された場所に物理的に移動しなければタスクを完了できない場合が想定される。この場合については、ワーカがタスクを行うことができる最大許容数と最大許容半径を定義することで、ワーカ対タスクの「最大割当て問題」を「最大フロー問題」に変換して解いた非特許文献3の研究が存在する。

【0009】

この技術は、1時点のワーカおよびタスクの情報を用いてグラフを作成し、最適化計算をおこなっているため、時刻によってワーカ集合やワーカによるタスク完了のコストが変化する場合を同時に扱うことができず、1時点ずつ順番にタスク割当てを行う貪欲法をとっている。

30

【先行技術文献】

【非特許文献】

【0010】

【非特許文献1】Dinitz, Y., Garg, N., & Goemans, M. X. (1999). On the single-source unsplittable flow problem. *Combinatorica*, 19(1), 17-41.

【非特許文献2】“クラウドソーシングなら日本最大級の「ランサーズ」” [online], 平成26年10月21日検索, インターネットURL<<http://www.lancers.jp/>>

【非特許文献3】Kazemi, L., & Shahabi, C. (2012). Geocrowd: enabling query answering with spatial crowdsourcing. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems* (pp. 189-198).

40

【発明の概要】

【発明が解決しようとする課題】

【0011】

しかしながら、非特許文献3の技術を用いて複数時点のタスク割当てをタスク完了時刻が最速になるように行う際、タスク完了時刻をコストとして「最小コストフロー問題」を解いても1時点分の最適解しか得られない。

【0012】

したがって、時刻によってワーカ集合やワーカによるタスク完了のコストが変化する場合を同時に扱うことができないおそれがある。また、1時点ずつ順番に最適化を行う貪欲法では、必ずしも時間的な大域的最適解は得られず、局所最適解しか求められていないお

50

それもある。

【0013】

本発明は、このような従来の問題を解決するためになされ、複数時点での最適化を1度に行って時間方向での大域的最適解を得ることを解決課題としている。

【課題を解決するための手段】

【0014】

本発明のタスク割当て装置は、時刻IDとワーカIDとタスクIDとが対応付けられた第1データベースと、タスクが発生する位置情報が格納された第2データベースと、第1データベースからワーカIDとタスクIDのペアと、該ペアの時刻IDとを抽出する手段と、抽出されたタスクIDをキーにして第2データベースからタスクの位置情報を検索する手段と、抽出されたワーカIDをキーにしてワーカの時刻毎の位置情報が格納された第3データベースから時刻毎のワーカの位置情報を検索する手段と、時刻IDが示す時刻毎にワーカがタスクの発生する位置に移動した場合の時刻を算出する手段と、算出された時刻のうち最も早い時刻と、該時刻に対応するワーカとタスクと該時刻の時刻IDとを選択して出力する手段と、を備える。

10

【0015】

本発明のタスク割当て方法は、時刻IDとワーカIDとタスクIDとが対応付けられた第1データベースに基づきワーカIDとタスクIDのペアと、該ペアの時刻IDとを抽出する第1ステップと、抽出されたタスクIDをキーにしてタスクが発生する位置情報が格納された第2データベースからタスクの位置情報を検索する第2ステップと、抽出されたワーカIDをキーにしてワーカの時刻毎の位置情報が格納された第3データベースから時刻毎のワーカの位置情報を検索する第3ステップと、時刻IDが示す時刻毎にワーカがタスクの発生する位置に移動した場合の時刻を算出する第4ステップと、算出された時刻のうち最も早い時刻と、該時刻に対応するワーカとタスクと該時刻の時刻IDとを選択して出力する第5ステップと、を有する。

20

【0016】

なお、本発明は、前記タスク割当て装置としてコンピュータを機能させるプログラムの態様としてもよい。このプログラムは、ネットワークや記録媒体などを通じて提供することができる。

【発明の効果】

30

【0017】

本発明によれば、複数時点での最適化を1度に行って時間方向での大域的最適解を得ることができる。

【図面の簡単な説明】

【0018】

【図1】2部グラフの変換例。

【図2】本発明の実施形態に係るタスク割当て装置の構成図。

【図3】割当て検索機能部の処理を示すフローチャート。

【図4】コスト計算機能部の処理を示すフローチャート。

【図5】制約式計算機能部の処理を示すフローチャート。

40

【図6】2部グラフにおける各辺の容量制約例。

【図7】「+」、「-」の説明図。

【図8】割当て最適化機能部の処理を示すフローチャート。

【発明を実施するための形態】

【0019】

装置構成例

図2に基づき本発明の実施形態に係るタスク割当て装置の構成例を説明する。このタスク割当て装置001は、クラウドソーシングにおけるワーカ対タスクの割当て問題の解決を図っている。

【0020】

50

具体的にはタスク割当て装置 0 0 1 は、コンピュータにより構成され、CPU、主記憶装置（RAM、ROM等）、補助記憶装置（ハードディスクドライブ装置、ソリッドステートドライブ装置等）などのハードウェアを備える。

#### 【0021】

このハードウェアリソースとソフトウェアリソース（OS、アプリケーションなど）との協同の結果、タスク割当て装置 0 0 1 は、時刻間隔 DB 0 0 5、タスク DB 0 1 0、ワーカ DB 0 2 0、割当て候補組検索機能部 0 3 0、割当て候補組 DB 0 4 0、コスト計算機能部 0 5 0、コスト DB 0 6 0、制約式計算機能部 0 7 0、制約式 DB 0 8 0、割当て最適化計算機能部 0 9 0、割当て組 DB 1 0 0 を実装する。

#### 【0022】

この各 DB 0 0 5、0 1 0、0 2 0、0 4 0、0 6 0、0 8 0、1 0 0 は、前記記憶装置に構築されているものとする。なお、タスク割当て装置 0 0 1 は、単一のコンピュータに構成してもよく、あるいは複数のコンピュータに前記各部 0 1 0～1 0 0 を分散して構成してもよいものとする。以下、前記構成要素 0 1 0～1 0 0 の処理内容を説明する。

#### 【0023】

時刻間隔 DB 0 0 5、タスク DB 0 1 0、ワーカ DB 0 2 0

#### 【0024】

#### 【表 1】

時刻間隔 ID	時刻
1	2014-02-02 06:00
2	2014-02-02 07:00
3	2014-02-02 08:00
4	2014-02-02 09:00
5	2014-02-02 10:00
...	...

#### 【0025】

表 1 は、時刻間隔 DB 0 0 5 のデータ構造例を示している。この時刻間隔 DB 0 0 5 には、ワーカとタスクの割当て間隔の情報が「YYY-MM-DD」の時刻形式で格納されている。例えば表 1 では、2014 年 2 月 2 日の時刻情報が 1 時間おきに格納されている。なお、時刻間隔 DB 0 0 5 は、外部からレコードを指定するために、各レコードに時刻間隔 ID が付与されている。

#### 【0026】

#### 【表 2】

タスク ID	タスク 緯度	タスク 経度
1	35.67661	139.76119
2	35.67661	139.76120
3	35.67661	139.76121
4	35.67661	139.76121
5	35.69805	139.77001
...	...	...

## 【 0 0 2 7 】

表 2 は、タスク D B 0 1 0 のデータ構造例を示している。このタスク D B 0 1 0 には、タスクの緯度経度の情報が格納されている。なお、タスク D B 0 1 0 も、外部からレコードを指定するために、各レコードにタスク I D が付与されている。

## 【 0 0 2 8 】

## 【表 3】

ワーカ ID	時刻	ワーカ 緯度	ワーカ 経度	最大 移動距離	最大 受容 タスク数	移動 手段
1	2014-02-02 06:00	33.67661	139.76119	120	2	Walk
1	2014-02-02 07:00	35.67661	139.76120	1200	2	Walk
1	2014-02-02 08:00	35.67661	139.76121	60	2	Car
2	2014-02-02 06:00	35.67661	139.76121	250	3	Car
2	2014-02-02 07:00	35.69805	139.77001	10	3	Bus
...	...	...	...	...	...	...

10

## 【 0 0 2 9 】

表 3 は、ワーカ D B 0 2 0 のデータ構造例を示している。このワーカ D B 0 2 0 には、  
ワーカの時刻毎に緯度経度情報と最大移動距離情報と最大受容タスク数の情報と移動手段  
情報とが格納されている。

20

## 【 0 0 3 0 】

表 3 中の最大移動距離は、ワーカがその時刻にタスクをするために移動できる最大距離  
を示している。また、最大受容タスク数は、ワーカが行うことができる最大のタスク数を  
示し、時刻毎に変化することなく一人のワーカには同一の値が付与されて格納されてい  
る。

## 【 0 0 3 1 】

さらに移動手段は、徒歩や車あるいはバスなどの移動手段情報を示している。ここでは  
ワーカの将来の場所や最大移動距離、移動手段がワーカによって既に入力され、それらの  
情報が既知であることを想定している。あるいはワーカの過去のログを利用して将来のス  
ケジュールを推定していることを想定している。なお、ワーカ D B 0 2 0 も、外部からレ  
コードを指定するために、それぞれのレコードにはワーカ I D が付与されている。

30

## 【 0 0 3 2 】

割当て候補組検索機能部 0 3 0 , 割当て候補組 D B 0 4 0

割当て候補組検索機能部 0 3 0 は、前記 D B 0 0 5 , 0 1 0 , 0 2 0 の格納データを入  
力とし、その処理情報を割当て候補組 D B 0 4 0 に出力して格納させる。ここでは割当て  
候補組検索機能部 0 3 0 は、前記 D B 0 0 5 , 0 1 0 , 0 2 0 の格納データに基づき割当  
てが可能な「時刻, ワーカ, タスク」の組(ペア)を検索し、検索結果を割当て候補組と  
する。

40

## 【 0 0 3 3 】

以下、図 3 に基づき割当て候補組検索機能部 0 3 0 の処理内容を説明する。まず処理を  
開始すると集合 N を用意して空集合とする ( S 1 - 1 )。その後時刻間隔 D B 0 0 5 か  
らレコード集合を検索して検索結果を集合「 H 」とし ( S 1 - 2 )、集合 H から未処理の  
要素を取り出して取り出した要素を「 h 」とし ( S 1 - 3 )、タスク D B 0 1 0 からレコ  
ード集合を検索して検索結果を集合「 T 」とする ( S 1 - 4 )。

## 【 0 0 3 4 】

つぎにワーカ D B 0 2 0 から時刻が要素「 h 」と一致するレコード集合を検索して検索  
結果を集合「 W 」とし ( S 1 - 5 )、該集合「 W 」から未処理の要素を取り出して「 w 」  
とする ( S 1 - 6 )。

50

## 【 0 0 3 5 】

続いて要素「w」の緯度経度を中心として、最大移動距離を半径とする円の内側にある座標を緯度経度とする「T」を検索する。これにより「時刻，ワーカ，タスク」の割当て候補組が特定され、特定された割当て候補組の各ID（時刻間隔ID，ワーカID，タスクID）を集合「N」に追加する（S1-7）。

## 【 0 0 3 6 】

この処理は、ワーカの指定する距離条件にタスクの位置が該当するか否かを確認するため、円の内外判定でもよく、あるいは最大移動距離を対角線とする矩形の内外判定などでもよい。

## 【 0 0 3 7 】

そして、未処理の要素「w」が集合「W」に存在するか否かを確認し（S1-8）、存在すればS1-6に戻る一方、存在しなければS1-9に進む。このS1-9では、未処理の要素「h」が集合「H」に存在するか否かを確認し、存在すればS1-3に戻る一方、存在しなければS1-10に進む。このS1-10では、集合Nの情報を割当て候補組DB040に出力して格納させ、処理を終了する。

## 【 0 0 3 8 】

## 【表 4】

時刻間隔 ID	ワーカ ID	タスク ID
1	1	1
1	2	1
2	1	1
2	2	2
3	1	1
4	2	3
4	3	4
...	...	...

## 【 0 0 3 9 】

表4は、割当て候補組DB040のデータ構造例を示している。この割当て候補DB040には、割当て候補組の時刻間隔ID，ワーカID，タスクIDが格納されている。

## 【 0 0 4 0 】

コスト計算機能部050，コストDB060

コスト計算機能部050は、割当て候補組DB040の格納データを入力として、その処理データをコストDB060に出力して格納させる。具体的にはコスト計算機能部050は、割当て候補組の「ワーカ，タスク」のペアのうち最小タスク完了時刻をとるものを複数時刻から選択して、コスト情報としてコストDB060に出力する。以下、図4に基づきコスト計算機能部050の処理内容を説明する。

## 【 0 0 4 1 】

S2-1：まず、処理が開始されると集合Cを用意する。この段階の集合Cは空集合とする。

## 【 0 0 4 2 】

S2-2：つぎに割当て候補組DB040を読み込み、未処理の「ワーカ，タスク」のペアを取り出して複数時点分の集合「P」とする。すなわち、複数時刻分の未処理の「ワーカ，タスク」ペアを集めて集合Pとする。

## 【 0 0 4 3 】

この処理を表4のデータ例に基づき説明する。ここでは時刻を「」で表し、ワーカを

「w」で表し、タスクを「t」で表し、添え字をそれぞれのIDで表し、割当て候補DB 040において上の行から順に処理するものとする。

【0044】

したがって、未処理の「ワーカ，タスク」ペアとして $(w_1, t_1)$ が最初に選択され、複数時点分の集合「P」は「 $P = \{(1, w_1, t_1), (2, w_1, t_1), (3, w_1, t_1)\}$ 」となる。

【0045】

S2-3：集合PのタスクIDに一致するレコードをタスクDB 010から検索し、集合PのワーカIDに一致するレコードをワーカDB 020から検索し、最速タスク完了時刻「min\_time」を計算する。

10

【0046】

ここではタスクとして、指定された場所の写真を撮影するなどの少ない時間で終わらせるものを想定している。したがって、タスクに指定された場所に到着することは、タスク完了と同義あるいは近似したものとする。

【0047】

最速タスク完了時刻「min\_time」の計算を、「 $P = \{(1, w_1, t_1), (2, w_1, t_1), (3, w_1, t_1)\}$ 」に基づき説明する。まずワーカとタスクの緯度経度情報を用いて各時刻「1, 2, 3」における距離「 $dist_1, dist_2, dist_3$ 」を計算する。

【0048】

20

ここではワーカの移動手段は、ワーカによって既にタスク割当て装置001に入力され、ワーカDB 020に格納されていることを前提とする。そこで、ある時刻「 $k$ 」にワーカがタスクの位置までに前記移動手段で移動するのに必要な所要時間「 $duration_k$ 」は、式(1)により求めることができる。

【0049】

【数1】

$$duration_k = \frac{dist_k}{v_k}, \quad k = 1, 2, 3 \quad \text{式(1)}$$

30

【0050】

また、ある時刻「 $k$ 」にワーカがタスクの位置までに前記移動手段で移動した際の時刻「 $time_i$ 」は、所要時間「 $duration_k$ 」に移動開始時刻を加算すればよいので、式(2)で求められる。

【0051】

【数2】

$$time_k = \tau_k + duration_k, \quad k = 1, 2, 3 \quad \text{式(2)}$$

40

【0052】

その結果、ワーカがタスクを完了させる最も早い時刻「min\_time」は、式(3)に示すように、これらの最小値を取ることで求めることができる。

【0053】

【数3】

$$min\_time = \min(time_1, time_2, time_3) \quad \text{式(3)}$$

【0054】

50

S 2 - 4 : 時刻「min\_\_time」とワーカIDとタスクIDと時刻「min\_\_time」を取った時刻IDとを集合Cに追加する。なお、時刻「min\_\_time」を取った時刻IDが複数存在する場合は、一番早い時刻選択するなどのルールを設けることで集合Cに追加する時刻IDを一意に定めることができる。

【 0 0 5 5 】

S 2 - 5 , S 2 - 6 : 未処理の集合Pが存在するか否かを確認し ( S 2 - 5 )、存在すればS 2 - 2に戻る一方、存在しなければS 2 - 6に進む。S 2 - 6では、集合Cの情報をコストDB 0 6 0に出力して格納させ、処理を終了する。

【 0 0 5 6 】

このとき時刻「min\_\_time」は、コスト情報としてコストDB 0 6 0に格納される。なお、表5はコストDB 0 6 0のデータ構造例を示し、時刻IDとワーカIDとタスクIDとコスト情報が格納されている。

【 0 0 5 7 】

【表5】

時刻 ID	ワーカ ID	タスク ID	コスト
1	1	1	2
1	2	1	3
2	2	2	3
4	2	3	4
4	3	4	2
...	...	...	...

【 0 0 5 8 】

制約式計算機能部 0 7 0 , 制約式DB 0 8 0

制約式計算機能部 0 7 0は、割当て候補DB 0 4 0の格納データを入力とし、その処理情報を制約式DB 0 8 0に出力して格納させる。ここではワーカ対タスクの最大割当て問題を前述の最小コストフロー問題に置き換えるものとする。

【 0 0 5 9 】

具体的には制約式計算機能部 0 7 0は、図1で表されたグラフにおける容量制約式と流量保存制約式とを計算して出力する。以下、図5に基づき制約式計算機能部 0 7 0の処理内容を説明する。

【 0 0 6 0 】

S 3 - 1 : まず、処理が開始されると集合「R」を用意する。この段階の集合「R」は空集合とする。

【 0 0 6 1 】

S 3 - 2 : つぎに割当て候補組DB 0 4 0を読み込んで未処理の「ワーカ , タスク」ペアを取り出し、取り出されたペアを「P'」とする。この処理を表4のデータ例に基づき説明する。

【 0 0 6 2 】

ここでは割当て候補組DB 0 4 0の上の行から順に処理されるため、最初に処理される「ワーカ , タスク」のペアP'として「 $P' = \{w_1, t_1\}$ 」が取り出され、その後に「 $P' = \{w_2, t_1\}$ 」, 「 $P' = \{w_2, t_2\}$ 」, 「 $P' = \{w_2, t_3\}$ 」と順に取りだされる。

10

20

30

40

50

【 0 0 6 3 】

S 3 - 3 : 図 1 のような 2 部グラフのソース ( s r c ) から各ワーカを結ぶ辺について、ワーカの最大受容タスク数を上限とする容量制限制約式を作成し、作成した容量制限制約式を R に追加する。

【 0 0 6 4 】

この際、最適化計算を行うための決定変数として、「ノード m」から「ノード n」に流れる水の流量を「 $x_{m,n}$ 」とする。ここで容量制約は、各辺の流量が容量を超えないという制約を意味する。

【 0 0 6 5 】

すなわち、ソースから各ワーカについて、ワーカの最大受容タスク数を上限とする容量制限制約式を作成するとは、「ワーカ  $w_i$ 」の最大受容タスク数を「 $\max T_i$ 」とすると、式 ( 4 ) を作成することとなる。

【 0 0 6 6 】

【 数 4 】

$$x_{src,w_i} \leq \max T_i, w_i \in P' \quad \text{式 (4)}$$

【 0 0 6 7 】

例えば図 6 の 2 部グラフでいえば、式 ( 5 ) を作成することとなる。

【 0 0 6 8 】

【 数 5 】

$$\begin{aligned} x_{src,w_0} &\leq 2 \\ x_{src,w_1} &\leq 1 \end{aligned} \quad \text{式 (5)}$$

【 0 0 6 9 】

なお、この処理は非特許文献 1 と同様な処理である。

【 0 0 7 0 】

S 3 - 4 : 各タスクとシンク ( s i n k ) を結ぶ辺と各ワーカと各タスクを結ぶ辺について「 1 」を上限とする容量制約式を作成し、「 R 」に追加する。ここで S 3 - 3 と同様にして決定変数「 $x_{m,n}$ 」を用いると、各タスクとシンクを結ぶ辺と各ワーカと各タスクを結ぶ辺について「 1 」を上限とする容量制約式を作成するとは、式 ( 6 ) を作成することとなる。

【 0 0 7 1 】

【 数 6 】

$$\begin{aligned} 0 &\leq x_{t_j,sink} \leq 1, \quad t_j \in P' \\ 0 &\leq x_{w_i,t_j} \leq 1, \quad w_i \in P', t_j \in P' \end{aligned} \quad \text{式 (6)}$$

【 0 0 7 2 】

例えば図 6 の 2 部グラフでいえば、式 ( 7 ) を作成することとなる。

【 0 0 7 3 】

10

20

30

40

【数 7】

$$0 \leq x_{t_0, sink} \leq 1$$

$$0 \leq x_{t_1, sink} \leq 1$$

$$0 \leq x_{t_2, sink} \leq 1$$

$$0 \leq x_{w_0, t_0} \leq 1,$$

$$0 \leq x_{w_0, t_1} \leq 1,$$

$$0 \leq x_{w_1, t_1} \leq 1,$$

$$0 \leq x_{w_1, t_2} \leq 1,$$

式(7)

10

【0074】

なお、この処理は非特許文献3と同様な処理である。

【0075】

S3-5: ワーカ頂点とタスク頂点について流量保存則の制約式を作成し、「R」に追加する。流量保存則は、その頂点に流入する流量が流出する流量に等しくなるという制約を意味する。

【0076】

20

すなわち、ワーカ頂点とタスク頂点について流量保存則の制約式を作成するとは、図7に示すように「+」、「-」をそれぞれ頂点から流出する辺の終点集合・頂点へ流入する辺の始点集合とすれば、式(8)を作成することとなる。

【0077】

【数 8】

$$\sum_{v \in \delta^+(w_i)} x_{w_i, v} - \sum_{v \in \delta^-(w_i)} x_{v, w_i} = 0, \quad w_i \in P'$$

$$\sum_{v \in \delta^+(t_j)} x_{t_j, v} - \sum_{v \in \delta^-(t_j)} x_{v, t_j} = 0, \quad t_j \in P' \quad \text{式(8)}$$

30

【0078】

例えば図6の2部グラフでいれば、式(9)を作成することとなる。

【0079】

【数 9】

$$(x_{w_0, t_0} + x_{w_0, t_1}) - x_{src, w_0} = 0$$

$$(x_{w_1, t_1} + x_{w_1, t_2}) - x_{src, w_1} = 0$$

$$x_{t_0, sink} - x_{w_0, t_0} = 0$$

$$x_{t_1, sink} - (x_{w_0, t_1} + x_{w_1, t_1}) = 0$$

$$x_{t_2, sink} - x_{w_1, t_2} = 0$$

式(9)

40

【0080】

なお、この処理は非特許文献3と同様な処理である。

【0081】

50

S 3 - 6 : 割当て候補組 D B 0 4 0 に未処理のペア「P'」が存在するか否か確認する。確認の結果、存在すれば S 3 - 2 に戻る一方、存在しなければ S 3 - 7 に進む。

【 0 0 8 2 】

S 3 - 7 : 集合 R の情報を制約式 D B 0 8 0 に出力して格納させ ( S 3 - 8 )、処理を終了する。なお、表 6 は、制約式 D B 0 8 0 のデータ構造例を示し、最適化計算を行う際に利用する各制約式の情報テキスト形式で格納されている。この各制約式には、レコード検索のため、制約式 I D が付与されている。

【 0 0 8 3 】

【表 6】

制約式 ID	制約式
1	$x_{src,w_0} \leq 2$
2	$x_{src,w_1} \leq 1$
3	$0 \leq x_{t_0,sink} \leq 1$
...	...

10

20

【 0 0 8 4 】

割当て最適化計算機能部 0 9 0

割当て最適化計算機能部 0 9 0 は、コスト D B 0 6 0 および制約式 D B 0 8 0 の格納データを入力として、その処理情報を割当て組 D B 1 0 0 に出力する。具体的には割当て最適化計算機能部 0 9 0 は、制約式 D B 0 8 0 から与えられた制約式のなかで「最大フロー問題」を解き、その結果を利用して「最小コストフロー問題」を解く。図 8 に基づき割当て最適化計算機能部 0 9 0 の処理内容を説明する。

【 0 0 8 5 】

S 4 - 1 : 制約式 D B 0 8 0 からレコード集合を検索し、検索結果を集合「R'」とする。

【 0 0 8 6 】

S 4 - 2 : 「R'」の制約式による条件の元でソースから流出する最大流量を求め、「max F」とする。この最大流量「max F」は、前記決定変数より式 ( 1 0 ) のように求められる。

【 0 0 8 7 】

【数 1 0】

$$\max F = \sum_{v \in \delta^+ src} x_{src,v} \quad \text{式(10)}$$

30

40

【 0 0 8 8 】

この最大流量「max F」を求める最適化計算は、線形計画法で解くことができるし、また効率的なアルゴリズムも複数提案されているため、これらを利用してもよい (例えば非特許文献 1 など)。

【 0 0 8 9 】

S 4 - 3 : ソースから流出する流量上限とシンクへ流入する流量上限とを「max F」とする制約式を作成し、「R'」に追加する。ここでは式 ( 1 1 ) の制約式を作成すればよい。

50

【 0 0 9 0 】

【 数 1 1 】

$$0 \leq \sum_{v \in \delta^+ src} x_{src,v} \leq \max F$$

$$0 \leq \sum_{v \in \delta^- sink} x_{v,sink} \leq \max F \quad \text{式(11)}$$

【 0 0 9 1 】

S 4 - 4 : コスト D B 0 6 0 からレコード集合を検索し、コスト関数を作成する。コスト関数とは、最適化の目的関数となる割当て（ワーカ，タスク）間のコストの和のことである。すなわち、コスト D B 0 6 0 に格納されたワーカ集合とタスク集合とをそれぞれ「w」と「T」とし、ワーカ「w<sub>i</sub>」とタスク「t<sub>j</sub>」とのコストを「c<sub>ij</sub>」とすれば、式（12）を計算すればよい。

10

【 0 0 9 2 】

【 数 1 2 】

$$\sum_{w_i \in W} \sum_{t_j \in T} c_{ij} x_{i,j} \quad \text{式(12)}$$

20

【 0 0 9 3 】

ただし、S 3 - 4 の容量制約からワーカとタスク間の辺との容量は「1」以下とされる。また、コスト D B 0 6 0 には必ずしもすべてのワーカとタスクとのペアにおいてコストが計算されているわけではないが、コスト「c<sub>ij</sub> = 」とみなして十分おおきい数を代入すれば、計算することができる。さらに存在しないペアの場合は計算を抜かせば計算することができる。

S 4 - 5 : 「R'」の条件の元でコスト関数を最小化する。ここでコスト関数の最小化は、線形計画法で解くことができる。最小化した際に選択された「ワーカ，タスク」のペア集合、即ちワーカ・タスク間の決定変数「x<sub>m、n</sub>」が1以上の「ワーカ，タスク」のペア集合を「opt\_\_p」とする。

30

【 0 0 9 4 】

S 4 - 6 : コスト D B 0 6 0 から「opt\_\_p」を検索し、その時の時刻IDを含めた「時刻，ワーカ，タスク」のペアの情報（時刻ID，ワーカID，タスクID）を割当て組 D B 1 0 0 ・に出力して格納させ、処理を終了する。

【 0 0 9 5 】

【表 7】

時刻 ID	ワーカ ID	タスク ID
1	1	1
2	2	2
4	2	3
4	3	4
...	...	...

10

## 【0096】

表 7 は、割当て組 DB100 のデータ構造例を示している。ここでは最適化計算の結果、タスク完了までの時間の和が最小となる「時刻，ワーカ，タスク」のペアについての「時刻 ID，ワーカ ID，タスク ID」が格納されている。

20

## 【0097】

このようなタスク割当て装置 001 によれば、コスト計算機能部 050 の S2 - 3 において、割当て候補組 DB040 の割当て候補組「時刻，ワーカ，タスク」から取り出された「ワーカ，タスク」のペアのうち、複数時点のタスク完了時刻の最小時刻が計算されることから、最速タスク完了時刻をコストとするコスト関数を計算することが可能となる。

## 【0098】

その結果、複数時点での最適化を 1 度に行うことが可能となり、時間方向での大局的最適解を得ることができる。したがって、タスクに時間空間条件が付与されたクラウドソーシングサービスのような時間によってワーカ集合やワーカによるタスク完了時間が変化する場合であっても、タスク完了時刻が最小となる「ワーカ，タスク」ペアを選択でき、タスク完了時刻の最速化を実現することが可能となる。

30

## 【0099】

これにより少しでも早く仕事を終わらせたいと考えるタスク依頼者の満足度を向上させることができる。また、タスク割当て装置 001 は、特にワーカの将来のスケジュール把握や予測ができる環境では有意義である。

## 【0100】

## プログラム

本発明は、上記実施形態に限定されるものではなく、各請求項に記載された範囲内で応用・変形して実施することができる。例えば本発明は、タスク割当て装置 001 の各部 005 ~ 100 の一部もしくは全部として、コンピュータを機能させる文書検索プログラムとして構成することもできる。

40

## 【0101】

このプログラムによれば、S1 - 1 ~ S1 - 10，S2 - 1 ~ S2 - 6，S3 - 1 ~ S3 - 7，S4 - 1 ~ S4 - 6 の一部あるいは全部をコンピュータに実行させることが可能となる。

## 【0102】

前記プログラムは、Web サイトや電子メールなどネットワークを通じて提供することができる。また、前記プログラムは、CD - ROM，DVD - ROM，CD - R，CD - RW，DVD - R，DVD - RW，MO，HDD，BD - ROM，BD - R，BD - RE

50

などの記録媒体に記録して、保存・配布することも可能である。この記録媒体は、記録媒体駆動装置を利用して読み出され、そのプログラムコード自体が前記実施形態の処理を実現するので、該記録媒体も本発明を構成する。

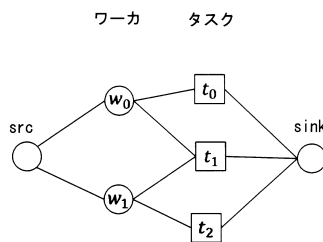
【符号の説明】

【 0 1 0 3 】

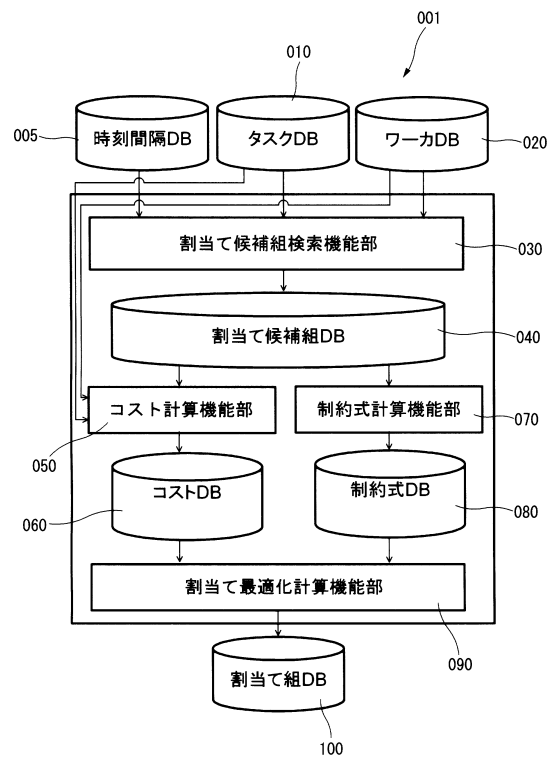
0 0 1 ... タスク割当て装置 0 0 1  
 0 0 5 ... 時刻間隔 D B  
 0 1 0 ... タスク D B (第 2 データベース)  
 0 1 0 ... ワーカ D B (第 3 データベース)  
 0 3 0 ... 割当て候補組検索機能部  
 0 4 0 ... 割当て候補組 D B 0 4 0 (第 1 データベース)  
 0 5 0 ... コスト計算機能部 0 5 0  
 0 6 0 ... コスト D B (第 4 データベース)  
 0 7 0 ... 制約式計算機能部  
 0 8 0 ... 制約式 D B (第 5 データベース)  
 0 9 0 ... 割当て最適化計算機能部  
 1 0 0 ... 割当て組 D B

10

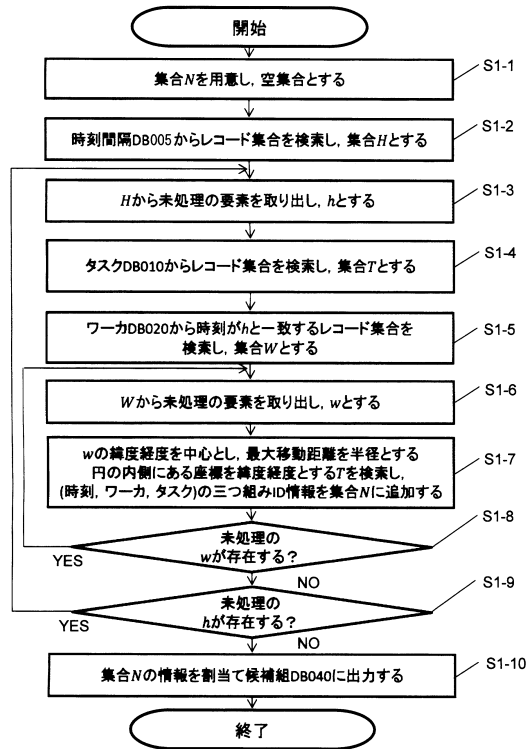
【 図 1 】



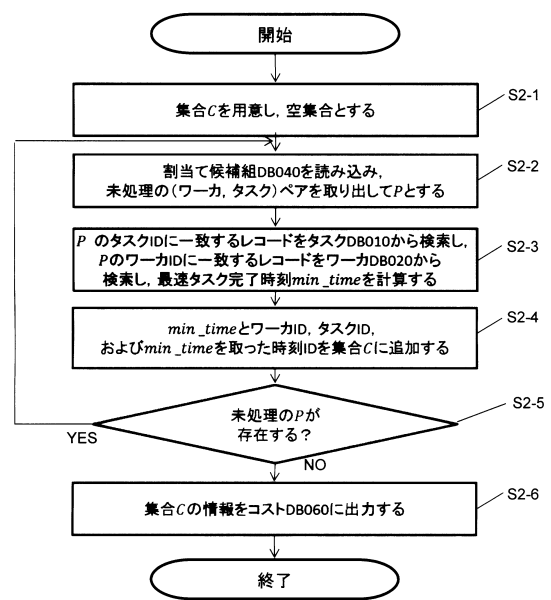
【 図 2 】



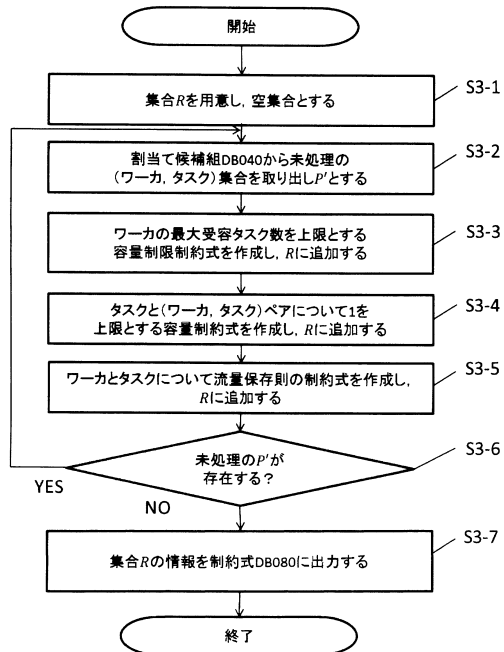
【図 3】



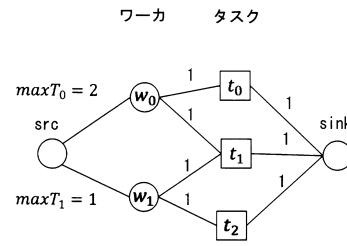
【図 4】



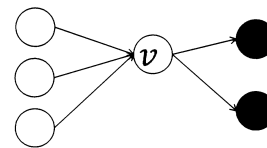
【図 5】



【図 6】



【図 7】



( $\delta + v$  は黒丸の頂点集合、 $\delta - v$  は白丸の頂点集合を表す)

【図 8】



---

フロントページの続き

- (72)発明者 戸田 浩之  
東京都千代田区大手町一丁目5番1号 日本電信電話株式会社内
- (72)発明者 小池 義昌  
東京都千代田区大手町一丁目5番1号 日本電信電話株式会社内

審査官 佐藤 裕子

- (56)参考文献 特開2014-179837(JP,A)  
特開2008-33433(JP,A)  
特開2012-3660(JP,A)  
特開2013-30127(JP,A)

- (58)調査した分野(Int.Cl., DB名)  
G06Q 10/00 - 99/00