本論文では、主にPTSPの変形例として、必要な人員数（シフト数）を最小化する方法に焦点を当てる。すなわち、始業時刻と終業時刻が設定された仕事と、それに対応する資格を持つスタッフ/機械/シフトの種類が与えられた場合に、使用するスタッフの数を2から制限せずに、総人数を最小化する解を見つけることに興味を持っている。このPTSPのバリエーションをシフト最小化PTSP（SMPTSP）と呼ぶ。このPTSPのバリエーションは、組織が持つべきスタッフの数や構成を決定するための戦略的な計画ツールとしても重要である。それは 日々の運用管理ツールとしても役立ちます。

第2節ではまず、本論文の関心事である問題を定義する。このセクションでは、いくつかの基本的な 表記法、問題の主要な特徴のいくつかを記述し、定式化を提供します。のSMPTSPの特性を説明する。2.4節でSMPTSPの特性を述べる。我々のヒューリスティックな セクション2のSMPTSPを解くためのアプローチを紹介する。我々のアプローチは、ボリューム アルゴリズムは3.1で説明したものに続いて、3.2で説明したWedelin アルゴリズムを使用します。我々は、我々の のアプローチを、ランダムに生成された2つのデータセットに適用した。これらのデータ、我々の詳細な計算解析と 我々のアプローチの計算性能に関する議論は、セクション4で提供される。

PTSPには、以下のような特定のモデルと目的関数に応じて、多くのバリエーションがあります。を選択します。残業代のみの最小化を選択する場合があります。私達は総数を最小にすることを選ぶかもしれません 使用されているスタッフの数、または使用されているスタッフの数によって発生する固定コスト。シフト最小化要員タスクスケジューリング問題(SMPTSP)では、私たちの目的は 使用される人員の総数を計算する。このPTSPのバリエーションは、以下のような場合に特に有用である。利用可能なカジュアルなスタッフの大規模なプールがあり、経営者は最小限に抑えたいと考えています。特定の日にこのプールから必要とされるスタッフの数を、その日のすべてのタスクのような方法で 日が適切に実行されます。利用可能なスタッフは、既知の開始時間との間でのみ働くことができます。マスター名簿、計画名簿、または情報を介して決定された既知の終了時刻 臨時職員名簿に記載されています。

ここでは、Krishnamoorthyら[14]ですでに提供されている表記を繰り返す。[14]. ここでは は以下のパラメータを定義します。

J 割り当てる必要のあるタスクのセット、J = {1, .... , n}.

W タスクを実行できるワーカーのセット、W = {1, ........ , m}.

W¯ 残業したいスタッフを表すWの部分集合

(これにはカジュアルなスタッフも含まれます。

が完全に残業していると判断された場合のコスト)。

ｃｊｗ タスクｊ∈Ｊを作業者ｗ∈Ｗが実行した場合のコスト

ｂｗ ワーカーｗ∈Ｗを使用した場合の固定費

sj タスク j ∈ J の開始時間。

fj タスク j ∈ J の終了時間（fj > sj ）。

Sw シフト開始時間ｗ∈Ｗ。

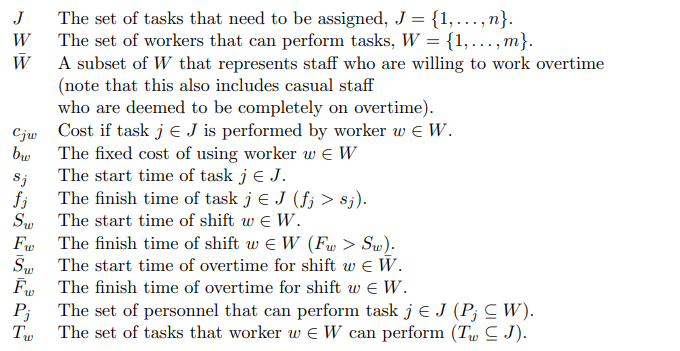
Fw シフトw∈Wの終了時間(Fw > Sw)。

S¯w シフトw∈Wの残業開始時間。

F¯w シフトｗ∈Ｗの残業終了時間。

Ｐｊ タスクｊ∈Ｊ（Ｐｊ⊆Ｗ）を実行できる要員の集合。

Ｔｗ 作業者ｗ∈Ｗが実行できるタスクの集合（Ｔｗ⊆Ｊ）。



また、上記のPTSPのすべての表記のリストは、完全性のためにここに含まれていることに注意してください。PTSPには使われているが、SMPTSPを記述する際に全てが使われているわけではないという意味では この論文でより詳細に説明しようとしている変種です。

上記の定義におけるシフトは、マシンやプロセッサを表すこともあります。S¯w = Sw または F¯w = Fw。また、カジュアルスタッフの場合は、S¯w = Sw、F¯w = Fwとなります。cjwは、一般的には、ゼロまたは 3 は比較的少ないです。これは、常勤のスタッフが常駐しているからです。は、その日に何をしたかに関係なく支払われる。しかし、コスト cjw > 0 が発生するのは 実際の状況では、W¯ ⊂Wは、タスクjを実行するためにカジュアルなスタッフや時間外スタッフを持ち込むことができます。掛け持ち に仕事をすることができます。

Krishnamoorthy et. 14]で示されているように、集合PjとTwは、時間窓 制約だけでなく、スキルの考慮も必要である。例えば，[sj , fj ]⊆ [Sw, Fw]があれば w ∈ Pj となり，ワーカー 集合Twは、集合Pjから、関係 Tw = {j | w∈Pj}. との昼休みは考慮していないことに注意してください。柔軟な開始時間と終了時間を指定することができます。ワーカー w を使用するために発生する固定費である bw を使用して、次のように指定することができます。ここで、作業者の中には、他の作業者よりもコストが高いものがあることを示している。ここで，区間グラフ G = (J, A) を定義してみよう．円弧集合である。この区間グラフでは、区間 [sj , fj ] と [sk, fk] が円弧で結ばれている場合、2つのノードj, k∈Jを円弧で結ぶ。が重なっている。言い換えれば、両方のタスクが同時に進行しているということであり、その意味では葛藤 との間にある。C をこの対立グラフの最大のクリオの集合とする．すなわち，C = {K1, ... , Kp}である．は、Kt の中の任意の 2 つのタスクがある時間間隔で重なるような集合 Kt ⊆ J で構成され、Kt でござる maximal. J \ Kt のタスクは、Kt のタスクと重ならない。 . すべての最大クリークを見つける を区間グラフで表現することは簡単である．我々は、すべての最大の 徒党。これがどのようにして行われるかの例を含め、詳細についてはKrishnamoorthy et. [14]を参照のこと。を用いて、PTSPを解くための可能なアプローチを定義する。ここでは、最大クリュークアルゴリズムを提供します。下

図

同様の問題は、文献では、固定ジョブスケジュール問題や異質なジョブスケジュール問題と呼ばれています。労働力スケジューリング問題 SMPTSPは、Valls et al. [17]のモデルに従う．労働力割り当て問題が研究されている．彼らは、以下を実行するために必要な労働者の数を最小化します。機械負荷計画。

SMPTSPの異なるバリエーションが文献に記載されています。すべての人員/機械が同一であれば とすべての作業が同じ種類の機械を必要とする場合（または、すべての労働者が同じスキルを持ち、すべての作業が同じ種類の機械を必要とする場合 作業者に同じスキルを要求する）とし、機械の数を最小限にするとしたら を使用した場合，結果として得られる問題は固定ジョブスケジュール問題(FJSP)である．この問題は Gertsbakh et. 10]およびFischetti et. [9]. FJSPは以下の方法でO(n log n)時間で解かれている. Gupta et al. 12] や Nakajima et. [16]. Arkin et. 1]は、O(n 2 対数n)アルゴリズム を使用する以外は，固定ジョブスケジュール問題に似た問題である．各仕事に何らかの価値を持たせた目的を持っています。言い換えれば、彼らはそれぞれの仕事がどのような問題を研究しているか は固定の開始時間と終了時間と値を持っています。そして、未完了のジョブの値を最小化します。

同一のマシン上で Arkin et. 1]は、固定ジョブスケジュール問題の彼らのバージョンが 各ジョブが処理可能なマシンのサブセットを持っている場合、NP-complete.

Fischetti et. [7]は，拡散時間制約を用いてFJSPを解く．このバージョンの問題では n 個のタスクで，それぞれ開始時間と終了時間 (sj , fj ) を持つ．また，無制限の数の同一の 常に利用可能なプロセッサ（Sw = -∞、Fw = ∞）を使用します。の数を最小化することが目的です。(SMPTSPのように)使用されているマシンの数を指定します．制約は，最後のジョブの終了時と終了時の差が 5 また、すべての機械の最初のジョブの開始は、最大のスプレッドタイムを超えないようにしてください。のための任意のアプローチ

SMPTSPはこの固定ジョブスケジュール問題を解決します。入力データを処理する。Fischettiら. [7] は，この問題がNP-hardであることを示している．

Fischetti et. ここでは、開始時刻と終了時刻を持つn個のタスクと無制限の数の同一のプロセッサ(Sw = -∞とFw = ∞)がある。目的は、与えられた作業時間T以上の間、どのプロセッサも合計では働かないように、最小限のプロセッサ数ですべてのタスクを完了させることである。tj = fj - sj とすると、SMPTSP と追加の の作業時間制約を用いて，上記の問題を導出する．我々の定式化では、SwとFwが有限であるため、各プロセッサ/人の作業時間は自動的に制約される。しかし、作業時間制約を使用することで、比較的少ないジョブ数のアプリケーションで作業負荷分散を達成することができるかもしれません。Kroon et. [15]は、戦術的固定間隔スケジューリング問題(TFISP)と呼ばれる問題を紹介しており、SMPTSPに最も近い問題かもしれません。彼らは、ジョブのセット(それぞれがジョブクラスに属し、開始時間と終了時間が固定されている)を非先制的に処理できる並列非同一マシンの最小数を決定しています。各マシンは、あらかじめ指定されたジョブクラスのサブセットからのタスクのみを実行することができ、一度に1つのジョブのみを処理することができる。問題TFISPは、SMPTSPとほぼ同じであるが、SMPTSPではマシンにも可用性の制限があることを除いては、SMPTSPと同じである。しかしながら、SMPTSPをTFISPとしてモデル化するには、全ての人員が、Swを処理すべき最初のジョブの開始点(∀w∈W)に等しく、Fwを処理すべき最後のジョブの終了点(∀w∈W)に等しく、固定時間のシフトで働くと仮定することで、SMPTSPをTFISPとしてモデル化することができる。さらに、労働者ｗの実際の利用可能時間ウィンドウの外にある開始時間および／または終了時間を有するすべてのジョブをＴｗから除外するために、入力データを前処理することができる。同様に、ジョブの完了時間ウィンドウの外にある実際のシフト時間を有するすべての要員をＰｊから除外するために、入力データを前処理することができる。

同様の問題は、Gondranら[11]で紹介されています。11] (476ページ、演習4)で紹介されています。この興味深い 変形演習は、すべてのフライトを飛行するために必要な最小限の航空機の数を識別することです。ある計画の地平線では，すべての飛行機が利用可能であると仮定しなければならない．のすべての時点ですべての飛行機が利用可能であると仮定しなければなりません。計画の地平線。言い換えれば、どの飛行機もメンテナンスや修理を必要とせず、また 同一平面による連続したフライト間のターンアラウンドタイムは無視できるほど小さい。また、我々は

地理的な接続性の制約を無視します。各フライト（私たちの場合はタスクに似ています）には 開始時間と終了時間が決まっていて、特定のタイプの飛行機（スタッフ、私たちの例ではスキル）を必要とします。

飛行機(シフト)が特定の時間帯に利用可能であることを考えると，我々は の飛行機(スタッフ)の数を計算する必要がある。したがって この問題は 本稿で検討している SMPTSP に似ています この 例は、Gondran et. 11]（204〜205頁、実施例23）に記載されている。Kroon et. 15]は、TFISPのいくつかの実用的な応用について議論しています。彼らは、TFISPの実用的な応用についても述べています。の複雑性を明らかにし、その解のための厳密・近似アルゴリズムを提供する。Kolen et. [13]は、間隔スケジューリング問題の包括的なレビューを提供している。特徴的なのは、仕事の期間とは別に、仕事の開始時間がまた持っているということです。誉められる この優れた論文は、結果の証明、複雑さの結果、近似性の証明を提供します。の結果が得られた。本論文では、以下のような興味深い応用例も紹介しています。本論文では、コテージレンタルや通信帯域の割り当てを含むインターバルスケジューリング問題のアルゴリズムを提供します。また、この論文では、間隔スケジューリングの2つのバリエーションのためのいくつかのアルゴリズムを提供しています。ＳＭＰＴＳＰである。最初のステップでは、各マシン（この例ではシフト）は、それが があります。第2の変形例は、すべての遊技機が継続して利用可能であるが、それぞれの遊技機が タスク/ジョブは、そのタスク/ジョブが処理できる最大の(または理想的な)マシンを持っています。SMPTSPは、実際には 考慮されてきた2つのバリエーションの組み合わせ、さらに、最小化を考慮しています。使用する機械/シフト数の合計の