マン・スケジューリングにおけるタブー探索法の適用性*

Adaptability of Tabu Search in Man Scheduling Problem *

奥谷 巖**・福井紀行***・風間克則****

*By Iwao OKUTANI ** ·Noriyuki FUKUI *** · Katunori KAZAMA ****

1．まえがき

先行後続関係を持つ複数の作業が1つのプロジェクトを構成しているものとしたとき，プロジェクトの納期と使用可能な資源（作業員や機械）の制限のもとで，経済効率的に望ましい各作業の開始時刻を求める問題をマン・スケジューリング問題というが時間をたとえば日単位といった離散時点で区切って扱かうと，典型的な組み合わせ問題となり，効率的な求解は一般には困難とされている。

この問題に対して，近年各方面で応用が盛んな遺伝的アルゴリズム（Genetic Algorithm, GA）を適用した解法については既に報告した[1)]。この方法の最大の特徴は計画目標を複数目標として設定できるという柔軟性にあり，この点が単一目標しか扱うことができない従来法とは大きく異なっている。

本研究は，マン・スケジューリング問題に対して一般にGAに比べて局所探索性能が高いとされているタブー探索法（Tabu search algorithm, TABU）[2)]を適用した方法について検討するものであるが，この方法においても計画目標は複数目標を指向する。したがって，具体的な計算例におけるアルゴリズムの性能評価はGAによる方法との比較対比をとおして行っている。

2．問題の設定

スケジュールを表現するネットワークはノードを作業に，アークを作業の先行後続関係に対応させた有向グラフで表わすものとし，ノードには作業番号i（i=0, 1, ･･･, n）を与えておく。ただし，始端ノード0と終端ノードnはダミーである。プロジェクトを構成する作業の技術的順序関係が与えられればこうしたネットワークは容易に構築することができるが，これを初期ネットワークとよぶことにする。なお，ここでは簡単のために単一資源の問題を扱かう。

さて，作業iの必要資源量a_iと処理時間b_iを与えPERT計算により初期ネットワークに対する作業iの最早開始時刻 t^E_i を求める[3)]。初期スケジュールとして各作業を最早開始時刻に開始するスケジュールを考えると，そのスケジュールのもとでのプロジェクト完了時刻T_{min}が求められるし，いわゆる山積み図を描くことで，初期スケジュールに対する最大投入資源量R^*が求められる。

以上のような準備のもとに，われわれはここで扱かう問題を次のように設定する。すなわち，まず納期 T_{max}と投入可能資源総量 R_{max}を与える。T_{max}はT_{min}を下回らない範囲で技術的判断によって与えればよいし，R_{max} は経済的判断に実行可能性の判断を加味して，たとえばR^* 前後の値として与えればよい。そうすると，問題は「投入資源をR_{max}以内に抑え，かつプロジェクトをT_{max}以内に完了するという条件のもとに，与えられた計画目標を最適化するスケジュールを求める」と記述できる。ここに，スケジュールを求めるということは，具体的には各作業の開始時刻t_iを決定することである。すなわち，納期T_{max}が与えられた段階で最遅開始時刻 t^L_iを求めたとき，計画目標が改善されるよう $[t^E_i, t^L_i]$ の範囲で t_i の値を定めるということである。

スケジュールの計画目標としては，プロジェクト完了時刻Tの最小化，最大投入資源量R（時刻kの投入資源量をR_k としたとき，その最大値）の最小化，

*キーワード：施工計画・管理

**正員，工博，信州大学工学部社会開発工学科
（長野市若里500，TEL026-226-4101，FAX 026-223-4480）
***正員，工修，名古屋市水道局
（名古屋市中区三の丸，TEL 052-62-6161，FAX 052-622-5941）
****非会員，信州大学工学部社会開発工学科
（長野市若里500，TEL 026-226-4101，FAX 026-223-4480）

遊休資源を少なくするための平滑度S（$=\sum R^2_k$）の最小化あるいは次式で与えられる資源使用効率Eの最大化が考えられる。

$$E=\sum_i a_i b_i / TR$$

こうした複数の目標を同時に考慮する簡単な方法の1つとして線形結合による和を考える方法があるがここでもその方法を採用する。Eを除く目標は無次元ではないので，まずそれらを次のような方法で無次元化し，f_iという1以下の非負数に変換する。

$$f_i=\frac{Y_{max}-Y}{Y_{max}-Y_{min}} \quad (i=1,2,3)$$

ここに，Yはi=1, 2, 3の順にT, R, Sの値をとるものとする。また，f_4=Eとしておく。このとき，スケジュールの目標Fを

$$F=\sum_{i=1}^{4} \alpha_i F_i \qquad (1)$$

とし，この最大化を図る。ただし，α_iは和が1となる各目標のウェイトである。

3．TABUの適用法

（1）TABU 探索法

TABU探索法は最近提案された方法で，各種組み合わせ問題を中心に適用例が報告されている[4)-6)]。この最適化アルゴリズムは，探索過程におけるある1点からたとえ改悪であっても最良の近傍解に移動すること及び1度通った点を記憶しておいてその点への再帰を禁止することの2点に特徴があるが，後者の性質がタブーという独特の名前に繋っている。禁止条件は一般にタブーリストとよばれるところに記憶されるが，リストサイズと称する容量を設けておいて，禁止条件を適当な時間後に忘却するようにし，探索範囲が不必要に狭められないように工夫している。

（2）スケジュールのビット列表現[1)]

初期スケジュールは，すべての作業を最早開始時刻に開始させるようにしているため，Tの最小化という目標以外は無視された形となる。したがって，ここでは技術的な順序関係がない作業間に人為的な先行後続関係を生成させ，作業開始時刻を先送りすることによって計画目標の向上を図る。

いま，作業i, j間に技術的順序関係がないものとし，そこにたとえば作業iを作業jに先行させるという関係を発生させるとしよう。このことは初期ネットワークに対して，i→jのアークを1本追加することに他ならない。このことを次の記号δ_{ij}によって表わす。i<jとしたとき

$$\delta_{ij}=\begin{cases} 1: i\rightarrow j \text{ のアークを追加} \\ -1: j\rightarrow i \text{ のアークを追加} \\ 0: i \text{ と } j \text{ を結合せず} \end{cases} \qquad (2)$$

ところで，i，jを新たなアークで結合したとき

$$t^E_i + b_i > t^L_j \qquad (3)$$

となると，作業jを最遅開始時刻以前に始めることが不可能となるし，

$$t^L_i + b_i \leqq t^E_j \qquad (4)$$

となっていれば，作業jの先送りが不要となりアークの存在は無意味となる。

したがって(3)式または(4)式が成立する場合にはδ_{ij}=1を除外し，(3)式および(4)式でi，jを入れ替えた式を(3)'式，(4)'式としたとき，そのいずれか一方が成立する場合にはδ_{ij}=-1を除外して考える必要がある。また，(3)式と(4)'式がともに成立するか，(3)'式と(4)式がともに成立する場合には，δ_{ij}そのものを除いて考えなければならない。

このような検討を技術的順序関係がないすべてのノード間で行うことにより，アークを追加し得るノードペアー(i, j)と当該追加アークの性質を表わすδ_{ij}の取り得る値が決定される。δ_{ij}の値を1つのルールのもとに与え，それを適当に並べたいわゆるビット列をXとすると，このXによって新しいスケジュールが表現されることになる。

（3）スワップ操作

図-1に示した小規模プロジェクトを表わすネットワークを例に説明する。実線で示したアークは技術的順序関係を表わしているが，(2)で述べた方法によりビット列の要素を選んでゆくと

$$X=(\delta_{14}, \delta_{15}, \delta_{16}, \delta_{24}, \delta_{25}, \delta_{26}, \delta_{34}, \delta_{35}, \delta_{45}, \delta_{56})$$

となる。ここに，δ_{14}, δ_{15}, δ_{16} δ_{26}, δ_{56}は-1の値を，δ_{34}は1の値をそれぞれ取らない。図-1の破線のアークを初期ネットワークに付加した新たなスケジュールは

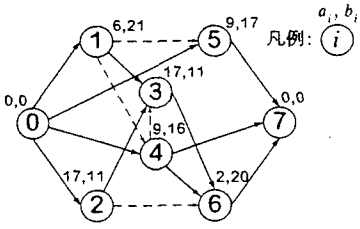

図-1 小ネットワーク

$$X=(1,1,0,0,0,1,-1,\ 0,0,0) \qquad (5)$$

のように書き表わされるが，この点（ビット列）の近傍解を求める操作を文献4）に倣ってスワップ操作とよぶことにする。

a ）スワップ 1

スワップ操作の１つの方法として，Xの任意の部分列 g の記号の順序を入れ換えるという方法が考えられる[7]。しかしながら，単純な順序の入れ換えだけでは，δ_{ij}のとり得ない値を与えるという不合理も発生するので，そうした場合には当該δ_{ij}の値を0にするという約束をしておく。例えば，(5)式で与えられるビット列 Xの第４要素から第７要素の部分列gに対して行うスワップ操作をSwap1(4,7)としたとき，操作前後のビット列は図-2(a)のようになる。スワップ操作後のネットワークの変化を具体的に記すと次のようになる。いまノードiからノードjに向かうアークをアーク(i,j)のように表わすものとしたとき

アーク(2,4)なし→新しいアーク(4,2)の生成

アーク(2,5)なし→新しいアーク(2,5)の生成

アーク(2,6)→消滅(アークの除去)

アーク(4,3)→消滅(アークの除去)

なお，gの順序の入れ換えによっても元のgと変わらないときは，当該スワップ操作は無意味であり実行しないものとする。

このようなスワップ操作をここではスワップ1とよぶ。

b ）スワップ 2

第2番目のスワップ操作（スワップ2）として，部分列gの各要素を一定の確率で別の値にするという方法が考えられる。いま，各要素を元の値のままに保持する確率をrとしたとき要素の変化し得る値が1つの場合は(1-r)の確率で，２つの場合は(1-r)/2の確率で別の値に切り変えればよい。(5)式のビット列Xのスワップ１と同じ部分列gに対して行う操作を

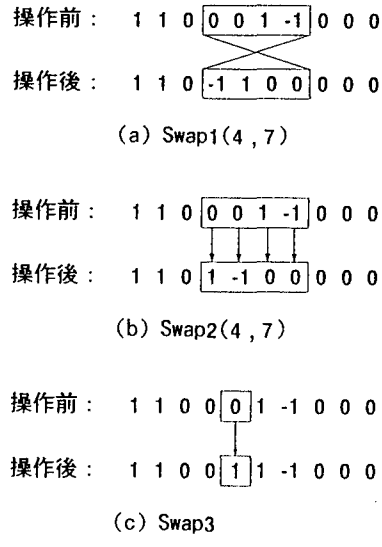

図-2 3種類のスワップ操作の模式図

Swap2(4,7)のように表わすものとしたとき，操作前後のビット列は図-2(b)のようになる。図の例では部分列の要素はすべて別の値に切り変わっているが，近傍探索というスワップ操作の本来の目的からすればrの値は大きめに設定し，gの要素の変化を低く抑えるのが妥当であろう。

ところで，スワップ1，スワップ2ともに部分列gのとり方を規定しておく必要があるが，ここでは次のような方法を採用した。すなわち近傍探索というスワップ操作本来の目的を逸脱しないようにし，かつ部分列の変更という操作の意味を損なわないようにするため，gの次元数n_gに対して

$$2 \leqq n_g \leqq Nq$$

なる制約を付しておき，この範囲の中でgの大きさをランダムに与えるようにする。ここに，Nはビット列Xの次元数，qは1未満の正の定数である。このようにして決められる部分列をXの左端から順次とってゆき，最終的にはXのビット列全体が部分列で被覆されるようにする。たとえば，Xがh個の部分列g_i(i=1～h)で覆われる場合

$$X=(g_1, g_2, \cdots, g_h)$$

となる。

c ）スワップ 3

第3番目のスワップ操作（スワップ3）として，Xの要素１つのみを別の値に変化させるという方法を考える。これはスワップ2でn_g=1,r=0とおいた特殊な場合であるとも解釈できる。

図－2(c)に(5)式のXの第5要素を0から1に変化させるスワップ操作を示したが，本スワップでは同じ第5要素を0から-1に変化させる操作は，計算過程の中では別のスワップとして扱う。

以上，3つのスワップ操作について述べたが，スワップ1はトラベリングセールスマン問題におけるスワップ操作[4]を模してはいるものの，物理的意味合いは薄い。スワップ2は部分列gの次元数が大きく，rが小さいと（要素を変化させる確率が大きいと），1回のスワップ操作でスケジュールは元のスケジュールからかなり離れてしまう可能性があり近傍探索の目的を遂行できなくなる恐れがある。最後のスワップ3は，最も近傍解の概念に近い解を探索すると思われるが，探索効率の点で若干弱さを持っている。

（４）タブーリスト

スワップ1,2では部分列の変化で近傍探索を実行しようとしていることを考え，いまから新たにスワップ操作を行おうとする部分列の中に，既に値を変化させた要素が含まれていればその部分列を禁止するという方法をとる。したがって，タブーリストの中味は変化させた要素の番号（Xの左端の要素から1,2,･･･のように付したもの）とすれば十分である。スワップ3については一旦その値を変化させた要素が元の値に戻ることを禁止するという方法をとることとし，タブーリストは値を変化させた要素番号と要素の元の値で構成するようにする。

なお，タブーリストにはそこに含めることのできる要素数の上限（＝タブーリスト・サイズL）を設けておく。要素数が上限に達している状態で新たに禁止すべき要素が現れた場合にはリストに組み入れた順番の古いものから消去してゆくようにする。

（５）計算手順

全体的な計算の流れは

①与えられたプロジェクトに対し，ランダムなビット列を複数個（K個）発生させる。

②それぞれのビット列を出発点として近傍探索を所与の回数（M回）繰り返えし局所最適解を求める。

③複数の局所最適解のうち最大の目標関数値を与えるスケジュールを最適解とする。

である。この中でランダムなビット列の発生については具体的には次のような方法を採る。初期スケジュールは工期の点で有利性を持っていることを考えると，ビット列X中の要素δ_{ij}の値の与え方として，その取り得る値を均等確率で割り振ることは得策ではない。つまり，Xの中に1や-1が多く含まれるようになれば，対応ノード間にアークが付加され，その都度関連作業の開始時刻が先送りされる可能性が高まって，結局，出発点となるスケジュールは既に工期の点でかなり不利な状態になっていることが考えられるからである。したがって，ここではδ_{ij}の与え方として0を取る確率をp(》0)，それ以外の値を取る確率を(1-p)(0以外の値が2つあれば均等確率で与える)として，乱数を用いて決定することとした。

以上のような準備のもとに，タブー探索法を用いた計算手順を示すが，スワップ操作の種類によらない共通的なアルゴリズム記述を期すため，スワップ3は1つの要素から成る部分列gの変化としてとらえる。

〈Step0〉 繰り返し回数K,Mおよびタブーリストサイズを与える。F^*_{best}=-1.0,k=1とおく。

〈Step1〉 ランダムなビット列Xを発生させ，PERT計算と山積み計算を行ってFを求める。タブーリストを空にし，m=1,F_{best} = Fとおく。

〈Step2〉 Xから部分列gを１つ取り出す。

〈Step3〉 部分列gにスワップ操作を施したとき$F \leqq F_{best}$ならば〈Step7〉へ。$F > F_{best}$ならば次のStepへ。

〈Step4〉 当該スワップを実行したとき、タブーを犯すならば〈Step6〉へ。そうでなければ次のStepへ。

〈Step5〉 $g^*=g$とし，gにスワップ操作を施したものをg^*_sとおき，さらに$F_{best}=F$として〈Step7〉へ。

〈Step6〉 $F > F^*_{best}$ならば〈Step5〉へ。そうでなければ次のStepへ。

〈Step7〉 与えられたビット列Xに対して，すべての部分列を調べ終えていれば次のStepへ。そうでなければ〈Step2〉へ。

〈Step8〉 Xのg^*の部分をg^*_sに置き換え，その結果得られた新たなビット列をXとする。

〈Step9〉 g^*とg^*_sの対応する要素の値が異なるも

のを取り出し，その要素をタブーリストに入れる。このとき，既にリスト中に記録されている要素と合わせて，その個数がLを上回われば，より記録時点の古い要素を消去し，リスト中の要素数をL個にする。

〈Step10〉 F_{best}〉F^*_{best}ならば$F^*_{best}=F_{best}$，$X^*=X$とおいて次のStepへ。そうでなければ何もせずに次のStepへ。

〈Step11〉 m<Mならばm:=m+1として〈Step2〉へ。m=M ならば次のStepへ。

〈Step12〉 k<Kならばk:=k+1として〈Step1〉へ。k=KならばX^*を最良スケジュールとして計算終了。

4．計算例による有効性の検討

(1)予備的実験

実際に計算を行うにあったては，割合または確率q,r,pの値およびL,K,Mの値を何らかの方法で与える必要があるが，ここでは試行錯誤法によってそれらの値を１つずつ決めてゆく。後のGAとの有効性比較においては，ノード数8～70のネットワークを対象としていることを考え，中間的大きさのノード数38のネットワークを予備検討の主たる対象として具体的な実験を実行する。当該ネットワークを含め本研究では実験対象となるプロジェクトはすべて計算機の中で生成（各作業の処理時間，必要資源量，作業間の技術的順序関係を乱数を用いて決定）するという方法を採っている。なお，目標関数Fを決めるウェイトα_1～α_4はそれぞれ0.3,0.4,0.3,0とした。

ａ）qの決定

スワップ1,2における部分列gの大きさの上限を規定するqの決定にあたっては，r=0.5,p=0.99,L=0.1N（整数），K=5,M=30と置き，q=0.01～0.1の範囲で0.01きざみに変化させるという実験を行った。スワップ1では0.05,スワップ2では0.01が最良値であるという結果が得られたが，スワップ2における0.01という値は，ノード数30以下のネットワークであるとnとしてほぼ下限値2を与えてしまうという値であり，スワップ3との対比で考えた計算速度の優位性を弱める結果にも繋がりかねない。したがって，ここではスワップ2についても敢えてスワップ1と同じ0.05を採用することとしたが，このことによる目標関数Fの最大値の低下は3%程度に止まった。

ｂ）rの決定

スワップ2の部分列の要素を元の値のままに保つ確率rについては，q=0.05,その他の条件はa)と同じという状態で，0～0.9の範囲で0.1きざみで変化させてみた結果，r=0.8のときに目標関数が最も改善された。したがって，ここではこの値をrの値として固定することとした。

ｃ）pの決定

先にも述べたように，初期スケジュールを表わすネットワークにあまり多くのアークを付加しない形で，計算の出発点となるネットワークを与えるために，ビット列Xの要素δ_{ij}が0を取る確率pを大きめの値にしておく必要がある。r=0.8,それ以外はb)と同じ条件に保ち，p≧0.8の範囲で実験を行ってみた結果，スワップ1ではp=0.9が，他のスワップではp=0.99が最も良好な結果をもたらすことがわかった。したがって，以下ではこれらの値を採用することとした。

ｄ）タブーリストサイズLの決定

q,r,pの値は上で決定された値に固定し，K=5,M=30として，タブーリストサイズ決定のための実験を種々行ってみた。スワップ1,2の場合は，1回のスワップ操作ごとに複数の要素がタブーリストに入ってくる可能性があるため，調べるべきLの上限を予め知ることはできない。したがって，L=sN(0<s<1)と置いてsを0.01～0.5の範囲で適当に変化させながら検討を行ったが，結果的に顕著な傾向を把握できなかった。そこで，これらのスワップについてはc)以前の試行実験で既に用いているL=0.1Nをそのまま採用することとした。

スワップ3については，繰り返し計算の各段階で最大限１つの要素しかタブーリストに書き込まれないことから，Nの大きさのいかんに拘わらずL≦Mとなる。したがって，1≦L≦30の範囲でLを変化させる実験を行ってみたがL≧10でほぼ同程度の良好な結果が得られたので，ここでは最終的にL=30(=M)として与えることとした。

ｅ）繰り返し回数K,Mの決定

q,r,p,Lの値を既決定値に固定し，まずM=30と置いてKを1～20の範囲で適当に変化させてみたがスワ

ップ1,2については10以上のKで，スワップ3では5以上のKでほぼ同程度の良好な結果を生成することが判明した。よって，スワップ1,2に対してはK=10を，スワップ3についてはK=5をそれぞれ採用することとした。

次に，このようにして決められたKに対して，Mを10から50の間で変化させる実験を行ったが，スワップの種類によらずK≧30で結果の改善は飽和することがわかった。よって，Mの値として30を採用することにした。

なお，d)およびe)の実験についてはノード数20と30のネットワークに対しても実行したが得られた結論に有意な差はなかった。

以上，開発した計算アルゴリズム実験の段階において必要となる各種パラメータの与え方について述べた。パラメータのあらゆる組み合わせについて網羅的に調べてはいないので，本来の意味における最適設定とはならないが，一応の良好な組み合わせの候補の1つとしては位置づけられるであろう。以下においては，これらのパラメータを使ってスワップ操作種別相互の有効性比較とともに，既開発のGAを用いた方法との比較をとおした提案手法の有効性の検討を行う。

（２）有効性の比較検討

ランダムに作成したノード数8,20,30,38,50,60,70の合計7つのネットワークについて，タブー探索法でスワップ1を採用した方法（TABU1と略記），スワップ2を採用した方法（TABU2),スワップ3を採用した方法（TABU3)および遺伝的アルゴリズムを用いた方法(GA)の4方法による計算結果を示すと表-1のようになる。表中，上段の数値は得られた目標関数の最大値（正確には最良値）を表わしている。また，われわれの準備した計算プログラムでは，良好な代替スケジュールも生成するという視点から，計算途中で得られる解を，目標関数の大きい順に10個記憶しておくという方法を採っている。下段の数値はそれら10個の目標関数の平均値である。

表-1 ノード数による TABU と GA の比較

ノード数	8	20	30	38	50	60	70
TABU 1	0.7851	0.8085	0.5849	0.7850	0.7213	0.6868	0.7113
	0.5753	0.7957	0.5725	0.7677	0.7090	0.6588	0.7021
TABU 2	0.7851	0.8074	0.6516	0.7615	0.7415	0.6504	0.6521
	0.6447	0.8016	0.6347	0.7460	0.7331	0.6446	0.6484
TABU 3	0.7851	0.8402	0.6612	0.8215	0.8138	0.7611	0.7762
	0.6721	0.8350	0.6588	0.8209	0.8116	0.7609	0.7757
GA	0.7851	0.8097	0.6233	0.7449	0.7005	0.5838	0.7139
	0.6520	0.8066	0.6189	0.7385	0.6964	0.5834	0.7124

注） 上段：最大値，下段：上位 10 個の平均値

なお，GAにおける遺伝操作は既に行われた検討結果をもとに[8]，淘汰としてはエリート保存戦略を，交差には一様交差を，交配には期待値戦略を，突然変異には反転方式をそれぞれ採用するようにしている。

表-1より，まずTABUにおける３つのスワップ操作の有効性を調べてみると，スワップ1とスワップ2は押しなべて同等の性能を有していることおよびスワップ3が最大値，平均値ともに最も良好な結果を生成していることがわかる。しかしながら，計算時間についてはノード数が38を越えたネットワークではTABU1およびTABU2はいずれもTABU3の1/10～1/20程度となっており（計算機はHITAC M-860/60。TABU3のCPU時間：ノード数38で約100秒，ノード数70で約750秒），この点での相対的有利性は持っている。

次に，目標関数の改善の点で最も性能のよいTABU3とGAの比較を行うことにより，提案方法の有効性を検証することとするが，その場合，公平な比較条件を整えるため，GAにおける計算繰り返し回数を意味する世代数は，CPU時間がTABU3のそれとほぼ同程度になるように設定した。なお，2方法間に必要記憶容量上の有意な差はない。表から明らかなようにノード数8の最大値が同一である外はすべてTABU3の結果の方が優れている。われわれが先に報告した時点においては[9]，ノード数が多くなるとTABU3の対GA優位性は崩れたが，今回の計算プログラム改良によりこのような結果となっている。特に，出発点となるネットワークに対して付加アークを削除した効果は顕著であった。なお，図-3はノード数38のネッ

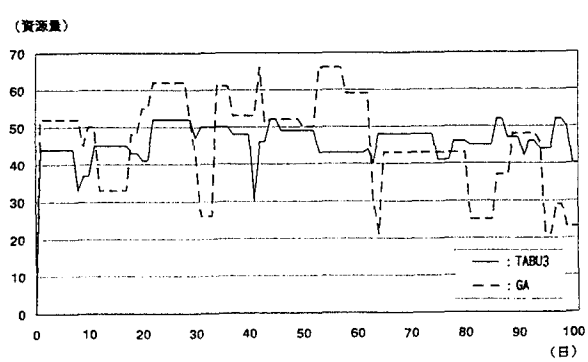

図-3 TABU3 と GA の山積み図比較（ノード数 38）

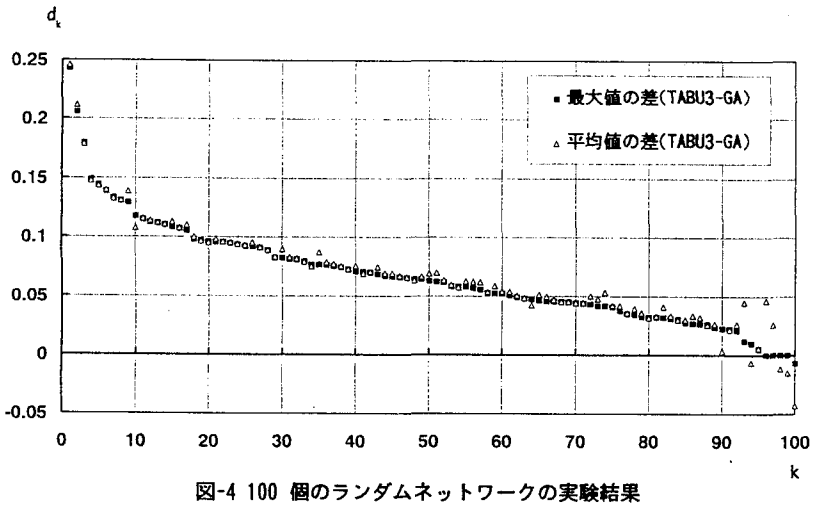

図-4 100 個のランダムネットワークの実験結果

トワークについて，TABU3とGAで求められた最大解の山積み図を示したものである。TABU3から得られた山積み図の方のより滑らかな資源量変化傾向が明らかに読み取れる。

GAについては世代数を多くとることにより，より優秀な個体（この場合は良好なスケジュール）が発生することが期待されるので，念のために世代数を10倍に増やした結果も求めてみた。しかしながら，結局TABU3を凌駕するまでには至らなかった。

上に述べた結果はあくまでも任意に選んだ7つの初期ネットワークについてのみ言えることとも考えられるので，より一般性を志向すべく，われわれは最後に100個のネットワークに対してGAとTABU3を適用するという実験を行った。ネットワークはノード数（=作業数）を8～88,作業iの必要資源量a_iを2～19,作業時間b_iを5～23日の範囲でランダムに与え，ノード間のアークも一定の条件の下にランダムに生成した。計算結果は図-4に示したとおりである。ただし，d_kはk番目スケジュール問題に対してTABU3で求めた目標関数とGAで求めた目標関数の差を表している。なお，kは最大値のd_kの大きさの順に設けた便宜的問題番号にすぎない。図より，最大値のd_kが1つの問題を除きすべて非負の領域にプロットされていることが読み取れるが，このことは実験の対象とした殆んど全てのネットワークについてTABU3の方がより良好なスケジュールを発見していることを意味している。

以上から，本研究で提案したタブー探索法を用いた方法（スワップ3使用）はGAを用いた方法に比べて有効性が高いと結論づけられる。

5．おわりに

本研究では，タブー探索法のマン・スケジューリング問題への適用に際して必須要件となる局所探索技法すなわちスワップ操作とそれに付随するタブーリスト形成法について検討し，計算例の中でGAとの比較をとおしてその有効性を調べた。その結果，初期ネットワークに対する付加候補リンクを逐次的に調べ上げるという基本原則に則ったスワップ3を使えば，安定的にGAを凌ぐ結果を生成することがわかった。

ここで示した方法はスケジューリングの複数の目標を同時に考慮し得る点に1つの特徴を有しているが，本稿ではその取り扱い方として加重和によるスカラ化手法を採用した。これによってパレート最適解の1つが得られるが，多目標の扱い方は多様であり，スケジューリングの方法としてはそうしたいろいろな意思決定の要請に沿う形でのアルゴリズムの弾力性を持っておいた方がよい。本研究で示した解法（GAもそうであるが）は，幸いというべきか，いわゆる制約条件および目標関数を陽表的に考慮する数理計画法ではなく，試行錯誤的に解を探ってゆく性質のものであるため，計算プログラムの若干の手直しにより，種々の多目標計画法の考え方にも対応し得る。

今後の方向として，広域的探索に力のあるGAの長所と局所探索を得意とするTABUを組み合わせ，さら

に計算効率を向上させてゆくことが考えられるが，現在までの試行ではあまりうまく行っていない。検討をさらに深め，将来的に稿を改めて発表したい。

参考文献

1)Huppe,B・奥谷　巖：マン・スケジューリング問題における遺伝的アルゴリズムの適用性，電気学会論文誌C,Vol.114-4,pp.450-455,1994.

2)北野：遺伝的アルゴリズム，産業図書，

3)須永：PERT系のプログラミング，朝倉書店

4)Malek,M.,et al:Serial and parallel simulated annealing and tabu sesrch algorithm for the travelling salesman problem,Annals of Operations Res.,Vol.21,PP.59-84,1989

5)Glover,F.:Tabu search-Part Ⅰ，ORSA J.on Computing,Vol.1-3,PP.190-206,1989.

6)Glover,F.:Tabu search-Part Ⅱ，ORSA J.on Computing,Vol.2-1,PP.4-32,1990.

7)奥谷　巖・Huppe,B・福井紀行：マンスケジューリングにおけるタブー探索法の応用，SICE学術講演会予稿集，PP.681-682,1994.

8)奥谷　巖・加藤正高・福井紀行：マンスケジューリングにおける遺伝的アルゴリズムの用い方，土木学会中部支部講演概要集，Ⅳ-16,1995.

9)奥谷　巖・福井紀行・風間克則：タブー探索法によるマンスケジューリングの解法，土木計画学研究・講演集，No.18(2),pp.401-404,1995.

マン・スケジューリングにおけるタブー探索法の適用性*

奥谷　巖**・福井紀行***・風間克則****

本研究では，マン・スケジューリング問題に対するタブー探索法（TABU）の適用方法が示されいる。提案方法は既に発表している遺伝的アルゴリズム(GA)による方法と同様に，1プロジェクトの複数目標を同時に考慮し得るという特徴を持っている。アローダイヤグラムで表わされた初期スケジュールに対して，新しい追加アークを示すビット列によって代替スケジュールを表わし，アルゴリズムを適用できるようにしている。局所探索技法であるスワップ操作について3つの方法が検討されているが，そのうちの最良のものを用いればGA法に比べて，より良好なスケジュールを生成し得ることを，ノード数8～88の100個のネットワークの計算例で実証している。

Adaptability of Tabu Search in Man Scheduling Problem *

*By Iwao OKUTANI** ·Noriyuki FUKUI*** ·Katunori KAZAMA*****

To apply the tabu algorithms schedule alternatives are expressed in the form of string whose elements indicate variation of affixation patterns of new arcs placed on the study scheduling network. Three swap operations (local search operations) are proposed and widely tested. Like the previously publised scheduling method utilizing genetic algorithms (GA), the proposed method has ability to find the optimum schedule which attains multiple goals of the project economy. Experimental results with 100 scheduling problems involving 8 to 88 activities show that the tabu method employing the best swap operation consistently outperforms the GA method.