実用的なジョブショップスケジューリング問題のための新しい遺伝表現とコモンクラスタ交叉

手塚 大[†] 樋地正浩[†]

この論文では実用的なジョブショップスケジューリング問題（JSSP）を最適化するための遺伝的アルゴリズム手法について述べる．この論文で提案する新しい遺伝子の表現方法とコモンクラスタ交叉（CCX: Common Cluster Crossover）は，納期遅れ最小化や段取り替え時間削減などの現実的なJSSPを最適化するのに適している．この手法では染色体上の遺伝子は作業順序の入替え情報を表す．CCXは2つの親染色体に共通なクラスタを交換して子孫を作る．この交叉によって，順列の中の部分列とその絶対位置を保存して子孫に伝えることができる．この手法を2つのベンチマーク問題と，音響部品メーカのスケジューリングに適用し，この手法が現実の問題において有効であることを確認した．

A New Genetic Representation and Common Cluster Crossover for Real-world Job Shop Scheduling Problems

Masaru Tezuka[†] and Masahiro Hiji[†]

This paper describes a genetic algorithm approach for real-world job shop scheduling problems. We developed a new genetic representation and an efficient crossover operator called Common Cluster Crossover (CCX). In our representation, chromosomes represent the shift of the order in a sequence. To preserve sub-sequences and their absolute positions in crossover operations, we implemented the process to identify the cluster of the sub-sequences and applied the CCX which exchanges common clusters between two parents. The application to scheduling of two benchmark problems and an audio parts manufacturer proved the effectiveness of the approach.

1. はじめに

ジョブショップスケジューリング問題（JSSP）とは，製造設備などのリソースに作業を割り当てていく問題である．ある目的のための作業の集まりをジョブと呼ぶ．ジョブによって製造に用いるリソースの使用順番が異なる製造形態をジョブショップと呼び，この製造スケジュールを立案する問題がJSSPである．実際の数多くの製造現場がジョブショップ形態であり，製造業において効率の良い生産を行うために，JSSPを最適化することが必要とされている．

遺伝的アルゴリズム（GA）は組合せ最適化問題の有効な近似解法と考えられ，Davisの研究[1]以来，多くの研究者が様々な方法でGAをJSSPに適用している．これらの研究にはJSSPを順序の最適化問題と考え，巡回セールスマン問題（TSP）で使われていた方法を適用する順序最適化アプローチ[2),3)]，作業の前後関係をビットで表現する方法[4)]，染色体に作業の開始時刻などを直接エンコードする方法[5)]などがある．

現実世界のJSSPの最適化では，段取り替え時間の削減や納期遅れの最小化などの複数の指標の最適化が要求される．ここで，段取り替え時間とは，ある作業を行うために製造設備の設定や補助具を変更するための時間である．同一種類の製品を連続して製造できると，設定の変更が不要となるため段取り替え時間を削減できる．したがって，段取り替え時間の最適化には，作業順の部分列が重要な意味を持つ．このほかに，食品加工では強い臭いや味のある食品を連続して加工したいという要望がある．また，薬品製造では，ある種の薬品を連続して加工することは危険があるため望ましくないという場合がある．これらの要望を考慮して最適化を行う場合にも作業順の部分列が重要となる．納期とは，その時間までに作業を終えなくてはならない日時・時刻である．納期より早く完成すると，発送日まで完成品を保管するためのコストが必要となる．

† 日立東北ソフトウェア株式会社研究開発部
Research and Development Division, Hitachi Tohoku Software, Ltd.

逆に納期遅れになると，顧客の信用を失ったり，ペナルティを課せられたりなど大きな損失となる．納期の最適化を行うには作業の順番の絶対位置が重要となる．同様に，中間工程の仕掛かり品の削減や，最終製品の安全在庫量の確保と過剰在庫の削減などの場合にも作業順の絶対位置が重要である．

以上のことから，実際の製造業における JSSP の最適化では，作業順の中の評価の良い部分列とその絶対位置を保存し，増やしていく必要がある．本論文ではこの 2 つの要求を満足する新しい GA 手法を提案する．2 章で，従来の順序最適化 GA 手法の JSSP への適用について述べる．3 章で，作業順の部分列とその絶対位置を保存し子孫に伝えて最適化を行うための新しい遺伝表現とコモンクラスタ交叉（CCX: Common Cluster Crossover）を提案する．4 章でシステムの構成を説明する．続いて 5 章で ft10x10 と ft20x5 のベンチマーク問題での評価実験と，実際の音響部品メーカの生産計画データを用いた評価実験の結果を示す．6 章は本論文のまとめである．

2. 従来の GA 手法の JSSP への適用

GA を JSSP に適用する場合，作業の重複や消失による実行不可能な作業順の生成を避けるための対策が必要である．

染色体に作業順を直接エンコードせず，染色体上の情報を実行可能な作業順に変換して用いる方法として Grefenstette ら[6] が TSP に適用した順序表現（Ordinal Representation）がある．この方法では遺伝子は残作業リストから次に取り出す作業の位置を示す．この方法には，通常の一点交叉や多点交叉が適用できるという利点がある．しかし，交叉によって作られる子の作業順は，親の特徴をほとんど受け継いでいない．

順序ベース交叉[2]，位置ベース交叉[2]，部分写像交叉（PMX）[7]，C1 交叉[8]，サイクル交叉[9]，エッジ再結合[10] などは，特別な交叉を用いることで，実行不可能な作業順の生成を避けている．これらの交叉法の最適化能力を比較した Starkweather ら[11] の研究によれば，エッジ再結合は TSP を最も効率良く解いたが，倉庫・発送スケジューリング問題では最下位であった．倉庫・発送スケジュールで最も成績が良かったのは位置ベース交叉であった．これはエッジ再結合が隣接する 2 つの要素間の関係のみに焦点をあてた交叉法であり，TSP のように都市を巡回する順番が逆になっても評価に影響ないような問題に有効であることを示している．しかし，倉庫・発送スケジューリング問題のように発送順が逆になると評価がまったく異なるような問題には有効であるとはいえない．このように，適用する問題に適切な交叉法を選択することは，問題を効率的に最適化するうえで重要な要素になる．

段取り替え時間や納期などを考慮しなければならない現実的な JSSP の最適化には，作業順の部分列とその絶対位置が重要である．この観点から従来の交叉法について考察を行う．

エッジ再結合は隣接する作業の前後関係のみが子孫に伝えられていき，作業の絶対位置は遺伝されない．これは，Starkweather らの研究で倉庫・発送スケジュール問題の最適化で最も成績が良くなかった原因の 1 つと思われる．サイクル交叉は両方の親の順序の絶対位置を受け継ぐが，作業の順序関係は受け継がない．このため，段取り替え時間の削減などには向かないと考えられる．PMX，位置ベース交叉，順序ベース交叉，C1 交叉は一方の親からは順序の部分列と絶対位置を受け継ぐが，もう一方の親からは相対的な前後関係しか受け継がない．このため，これら従来の交叉法では現実の JSSP の最適化能力はあまり高くないと考えられる．JSSP を効率良く最適化する交叉法としてサブシーケンス交換交叉（SXX）[12] が知られている．この交叉法は 2 つの親のサブシーケンスを構成する作業が集合として一致したときに，このサブシーケンスを交換する．この方法では両方の親の作業の順序関係を受け継ぐことはできるが，順列中の作業の絶対位置は受け継ぐことができない．そのため，SXX を現実の問題に適用する場合，順序関係を継承するという特徴から段取り替え時間の削減などの最適化に向いているが，作業の絶対位置を継承できないことから納期遅れや中間仕掛品の削減などには適さないと考えられる．

以上から，現実の JSSP に GA を適用するには，親世代が持つ作業順の，評価の良い部分列と絶対位置を次世代に伝えていく能力を持った新たな手法が必要になる．

3. 新しい遺伝表現と CCX

本章では，交叉時に，2 つの親の持つ作業順の部分列とその絶対位置を次世代に残すことを可能にするために，新しい遺伝表現とコモンクラスタ交叉（CCX）を提案する．

3.1 新しい遺伝表現

新しい遺伝表現では，染色体（chromosome）は作業順の入替え情報を表す．この情報を作業の順序に変換してスケジュールの作成を行う．全部で j 個の作業があるとすると，染色体の長さは $j-1$ となる．**図 1**

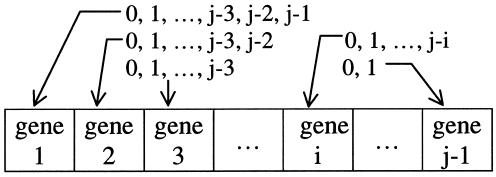

図 1 染色体上の遺伝子
Fig. 1 The alleles.

に示すように，i 番目の遺伝子（gene）の値は $[0, j-i]$ の整数値をとることができる．

染色体上の情報から作業順への変換の手順を**図 2** に示す．図 2 の例では染色体上の遺伝子は {5,4,1,0,1} である．

まず最初に作業（operation）の順番をなんらかの基準で整列する．図 2 では作業番号の昇順 {op1,op2, op3,op4,op5,op6} とする．遺伝子の 1 番目の値は 5 で，これは一番最初の作業の順番を 5 つ後にある作業と入れ替えることを意味する．そこで，作業 op1 と作業 op6 が入れ替わり，作業の順番は {op6,op2,op3,op4,op5,op1} となる．続いて，2 番目の遺伝子の値は 4 で，これは 2 番目の作業の順番を 4 つ後にある作業と入れ替えることを意味する．そこで作業 op2 と作業 op1 が入れ替わり，作業の順番は {op6,op1,op3,op4,op5,op2} となる．このように作業の入替えを繰り返すことで最終的に {op6,op1,op4,op3,op2,op5} という作業の順番を得る．つまり，染色体 {5,4,1,0,1} は作業順 {op6,op1,op4,op3,op2,op5} を表現している．この遺伝表現では，普通の一点交叉や多点交叉を適用しても，作業の重複や消失が発生しない．

3.2 コモンクラスタ交叉（CCX）

従来の GA では交叉点はランダムに選ばれてきた．本論文で提案するコモンクラスタ交叉（CCX）では，前節で説明した遺伝表現を用いたうえで，作業順の部分列と絶対位置を保存できるような位置を交叉点として選択する．CCX は部分列のクラスタ（cluster）を選び出し，2 つの親の染色体間で共通なクラスタを交換して交叉を行う．

クラスタとは作業順の入替え時に，相互に関係しあう遺伝子の集まりのことである．クラスタの例を**図 3** に示す．染色体上の各遺伝子の値は作業順の入替え情報である．図 3 の矢印は作業の入替え先を示している．この矢印で接続された遺伝子の集まりがクラスタである．図 3 では 1，2，8，9，10 番目の遺伝子の集まりが 1 つのクラスタ（Cluster 1）を構成し，3，4，5，6，7 番目の遺伝子の集まりがもう 1 つのクラスタ

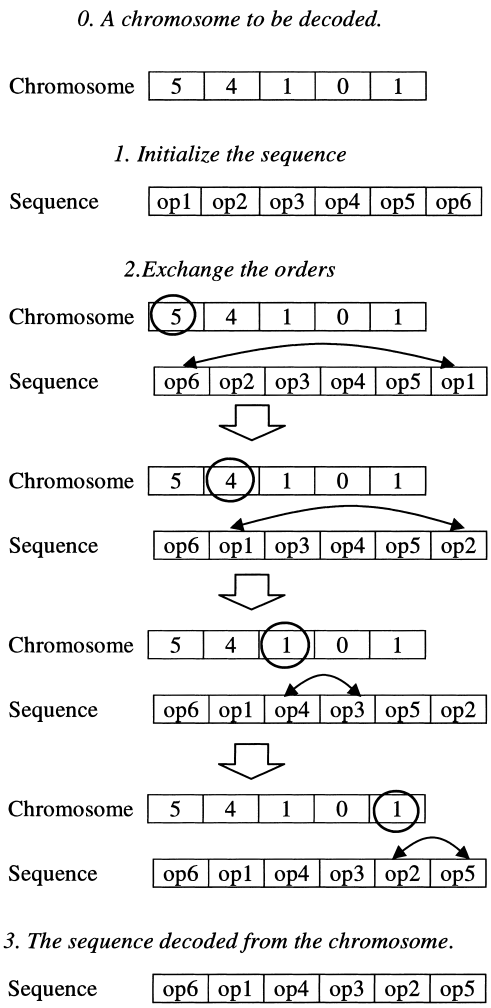

図 2 染色体上の遺伝子から作業順への変換
Fig. 2 Decoding process of the chromosome to the sequence.

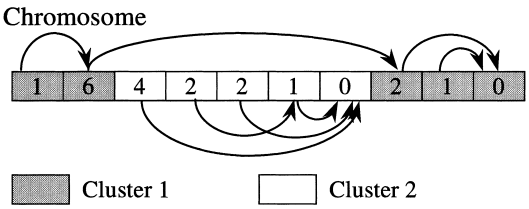

図 3 部分列のクラスタ
Fig. 3 The cluster of sequences.

（Cluster 2）を構成している．このようにクラスタには飛び地があってもよい．

CCX は 2 つの親の染色体の共通なクラスタを交換する．CCX でのクラスタの交換を**図 4** に示す．この例では親 1（Parent 1）の染色体は A と B の 2 つのクラスタからなり，親 2（Parent 2）の染色体は C，D，

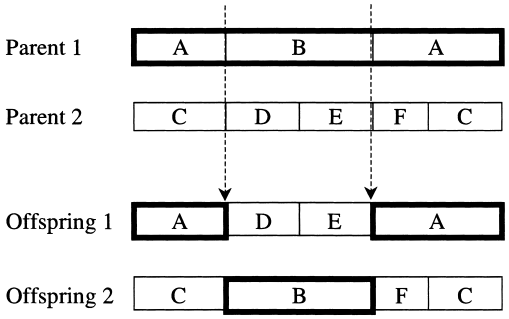

図 4 コモンクラスタ交叉（CCX）
Fig. 4 Common Cluster Crossover (CCX).

E，F の 4 つのクラスタからなる．2 つの点線矢印は，各クラスタの境界のうち，2 つの親で位置が一致する境界を示している．CCX は，このように両親で位置が一致する境界で区切られたクラスタを交換して 2 つの子（Offspring 1，2）の染色体を生成する．

図 5 は CCX の実行例を示している．この例では 2 つの親から生成される 2 つの子のうちの一方のみを例示している．親 1 の染色体は A，B の 2 つのクラスタを持ち，親 2 の染色体は C，D，E，F の 4 つのクラスタを持つ．2 カ所で両親のクラスタの境界が一致するので，ここを交叉点として親の染色体のクラスタの交換を行い，子の染色体を生成する．生成された子の染色体を作業順に変換すると，両方の親が持つ作業順の部分列と絶対位置を継承していることが確認できる．この図では親 1 からは {op2, op9, *, *, *, *, *, *, op10, op1,op11} という作業順を引き継ぎ，親 2 からは { *, *, op8, op6, op4, op5, op3, op7, *, *, *} という作業順を引き継いでいる．この結果，子の作業順は {op2, op9, op8, op6, op4, op5, op3, op7, op10, op1, op11} となる．このように，CCX を用いると，両方の親の部分列と絶対位置を子孫に伝えることができる．このことから，CCX は現実世界の JSSP を効果的に最適化できる．

ところで，CCX を適用できない場合が 2 つある．1 つは，2 つの親に共通するクラスタ境界が存在しない場合である．このような場合には，一方の親のクラスタに従って交叉を行い，もう一方の親のクラスタは無視する．このような交叉を疑似 CCX（Quasi-CCX）と呼ぶ．もう 1 つは，2 つの親とも 1 つのクラスタしかない場合，つまりクラスタ境界が存在しない場合である．このような場合には通常の一点交叉を行う．これらの疑似 CCX や一点交叉を行う場合には，作業の部分列や絶対位置の保存は保証されない．

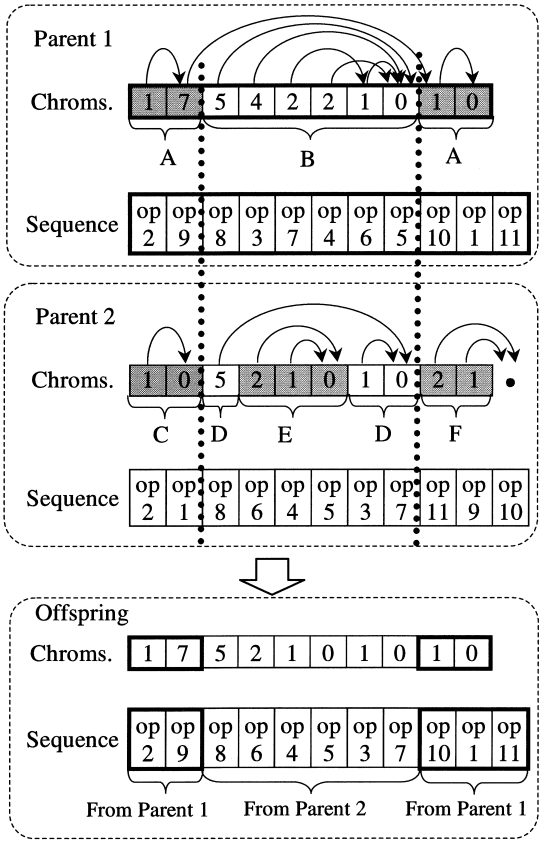

図 5 CCX の例
Fig. 5 An example of CCX.

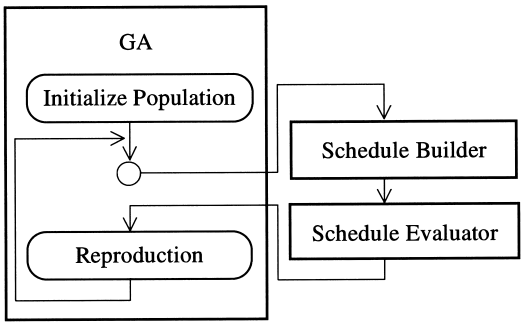

図 6 システムの構成
Fig. 6 GA scheduler.

4. システムの構成

本手法で用いたスケジューリングシステムの構成は**図 6** に示すように，GA 部，スケジューリング部（Schedule Builder），スケジュール評価部（Schedule Evaluator）から構成される．

GA 部は新しい遺伝表現と CCX によって作業の順序を最適化する．1 つの染色体は 1 つのリソース上の

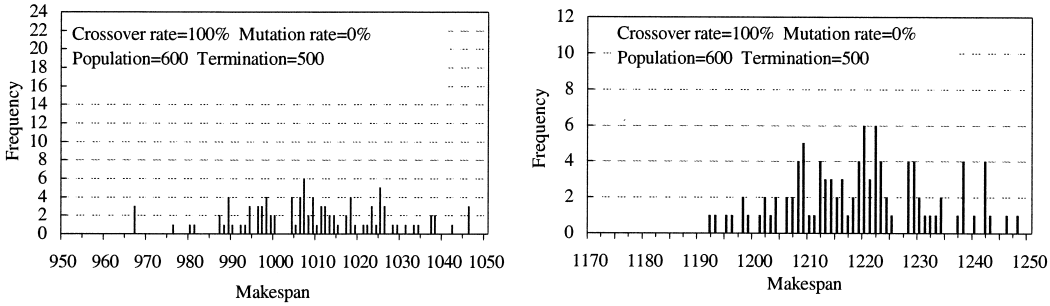

図 7 突然変異なし CCX による 100 試行中の Makespan の度数分布
Fig. 7 Frequency distribution of Makespan in 100 trials without mutation.

作業の順序を表し，1 つの個体は複数の染色体を持つ．スケジューリング対象となるリソースの数が，1 個体が持つ染色体の数となる．初期個体の生成（Initialize Population）では，染色体上の遺伝子の値を，$[0, j-i]$ の一様乱数整数で初期化する．ここで j は，対応するリソース上で処理される作業の数を表し，i は遺伝子の染色体上での位置を表す．突然変異では同じ範囲の一様乱数で遺伝子の値を変更する．

スケジューリング部は GA 部で生成された作業の順番をもとに，作業の前後関係などの制約条件などを考慮しながらスケジュールを作成する．

スケジュール評価部はスケジュール部で作成されたスケジュールを適合度に基づき評価する．この評価をもとに GA 部で選択・子孫の生成（Reproduction）が行われ，作業の順序の最適化が行われる．GA 部で行う子孫の生成では，早熟な収束を防ぐ効果のあるランク選択[13] と，評価の良い個体を次世代に残すエリート戦略[14] を採用した．

5. JSSP への適用

JSSP のベンチマークとして ft10x10[15]（10 ジョブ 10 マシン問題）と，ft20x5[15]（20 ジョブ 5 マシン問題）が広く用いられている．そこで，本論文で提案する新しい遺伝表現と CCX の有効性を評価するために ft10x10 と ft20x5 への適用，評価を行った．

ft10x10 と ft20x5 は最大完了時間（Makespan）の最小化を目的とし，各ジョブ中の作業の前後関係制約と，リソースは同時に 1 つの作業しか加工できないという JSSP の基本的な制約条件がある．しかし，実際の製造現場のスケジューリングに要求される納期や段取り替え時間などは問題に含まれていない．そこで，この点を補うために現実の JSSP の 1 つである音響部品メーカのスケジューリングへの適用，評価も行った．現実問題への適用では，ft10x10 で最も最適化能力が高いサブシーケンス交換交叉（SXX）[12] との比較を行った．

表 1 100 試行中の平均と標準偏差
Table 1 Average and standard deviation of 100 trials.

	ft10x10		ft20x5	
	average	stdev	average	stdev
CCX	1009.2	17.2	1219.1	12.7
SXX+GT [12]	934.3	5.8	1217.4	18.6
GTX [12),17]	967.7	7.1	1250.9	13.9

stdev: standard deviation

5.1 ベンチマーク問題での性能評価

スケジューリング部では，Giffler と Thompson によって考案されたアクティブスケジュールを生成するアルゴリズム[16] を用いてスケジュールを作成した．また，評価部で用いる適合度として Makespan を用いた．

(1) 交叉のみ（突然変異なし）

CCX の交叉性能のみを評価するために，交叉率を 100%，突然変異なしでスケジューリングを行った．エリート戦略のエリート数は 1 とした．試行の最良個体を，その試行の最終結果として採用した．解の分布を調べるために 100 試行を行った．集団サイズを 600 個体，世代交代数を 500 世代，すなわち総評価回数を 3.0×10^5 とした．Pentium III 550 MHz，メモリ 128 M バイト，Windows NT4.0 の PC 上で C++を用いて実装したシステムで 1 試行あたりの計算時間は約 5 分であった．

ft10x10 と ft20x5 の解の度数分布を**図 7** に示す．また，100 試行中の最良解の平均値（average）と標準偏差（stdev）を**表 1** に示す．比較のために小野らによる SXX と GT 法による強制の組合せ手法[12] と，Yamada らによる GT 交叉手法[17] の値も記載した．これらは小野らの報告[12] の値である．

小野らの報告では総評価回数を 3.0×10^6 としているが，本論文ではこの 10 分の 1 の 3.0×10^5 とした．これは CCX では交叉点がランダムに決まるのではなく，CCX によって決定されるため，集団内の多様性が失われると交叉点が限定され，早い段階で収束するためである．

ここで取り上げた 2 つのベンチマーク問題のうち ft20x5 のほうが難しい問題であることが知られている[18]．この実験で CCX は ft10x10 では他の 2 つの手法より劣るが，より難しい問題である ft20x5 では SXX と同程度，GTX よりも優れた結果を得ている．また，この値を得るのに要した総評価回数は SXX や GTX の 10 分の 1 である．ft10x10 でも少ない計算量で比較的良い解を得られている．

収束の状況を見てみると CCX はほとんどの試行で 200 世代以前に，またすべての試行が今回設定した 500 世代以前，つまり評価回数が 3.0×10^5 より少ない時点で収束していることが確認された．**図 8** は典型的な収束の様子を示している．

(2) 交叉と突然変異

続いて，突然変異ありでのスケジューリングを行った．交叉率を 80%，突然変異率を 20%とした場合の度数分布を**図 9** に示す．この実験では突然変異により早熟な過収束を避けられるため，世代数を 5000 とした．したがって 1 試行あたりの総評価回数は 3.0×10^6 である．その他の条件は前の実験と同じ，集団サイズ 600，エリート数 1 とした．

突然変異を使用した場合の ft10x10 での 100 試行中の最良解の平均値は 982.2，標準偏差は 15.5，ft20x5 では平均値は 1203.6，標準偏差は 8.5 であった．

CCX は，ft20x5 において少ない計算量で，SXX と同等，GTX よりも良い解を得られることが確認された．

5.2 音響部品メーカのスケジューリングへの適用

現実の JSSP における CCX の有効性を評価するために，典型的なジョブショップ生産を行っている音響部品メーカのスケジューリングに適用した．

このメーカは家電メーカの関連企業で，注文はすべてこの家電メーカから月次で受注する．したがって，生産スケジュールは月次で立案される．主要製品はスピーカとヘッドフォンで，生産規模は月産 7500 台である．MRP システムによってロットサイズが決定され，約 80 のジョブにまとめられる．リソースである製造設備は 15 あり，ジョブによって使用される製造設備が異なるジョブショップの製造形態をとる．1 カ月あたりの作業数は約 300 である．なお，今回の実験で用いたのはジョブ数 63，作業数 427 のデータである．

このメーカのスケジューリングの目的は納期遅れ，倉庫コスト，段取り替え時間の削減，つまり以下の式の最小化である．

$$
\begin{aligned}
f = {} & w_1 N_T \\
& + w_2 \sum_{j \in J} (\max(C_j - d_j, 0))^2 \\
& + w_3 \sum_{j \in J} \max(d_j - C_j, 0) \\
& + w_4 \sum_{r \in R} S_r \qquad (1)
\end{aligned}
$$

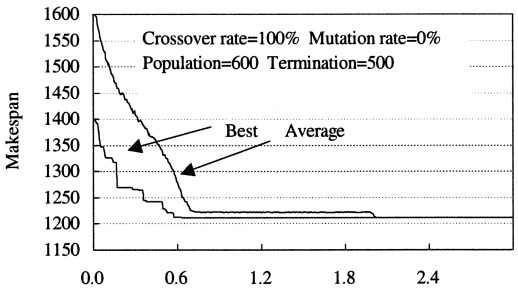

図 8 典型的な Makespan の収束の様子（ft20x5 の例）
Fig. 8 Typical temporal change of Makespan on ft20x5.

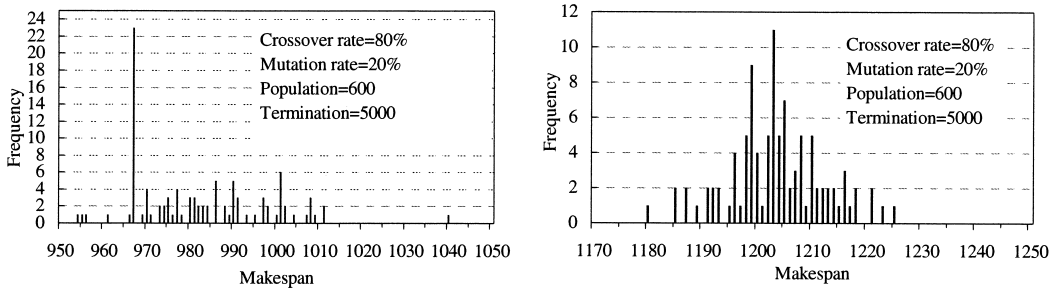

図 9 突然変異あり CCX による 100 試行中の Makespan の度数分布
Fig. 9 Frequency distribution of Makespan in 100 trials with mutation.

N_T は納期遅れジョブの数，J はすべてのジョブの集合，R はすべてのリソースの集合を表す．C_j はジョブ j の完了時刻，d_j はジョブ j の納期を表し，S_r はリソース r で段取り替えに要した時間を表す．

この式の第 1 項と第 2 項が納期遅れの削減を表す．第 1 項が納期遅れジョブの数を減らし，第 2 項は納期遅れが発生した場合にも納期遅れ時間ができるだけ短くなるようにする．第 3 項は倉庫コストの削減を表す．倉庫コストは納期より早く製造が完了したときに，完成製品を保管するための費用である．この倉庫コストを削減するためには，できるだけ完成時刻を納期に近づけるようにする．なお，第 2 項は納期遅れ，第 3 項は納期より早く製造が完了する場合で，どちらも完成時刻と納期のずれを表している．実務上，納期遅れを出さないことが優先されるため納期遅れのほうが二乗で効いてくるようにしてある．第 4 項は総段取り替え時間の削減を表す．段取り替え時間を減らすことにより製造設備の稼働率を向上させることができる．なお，w_1〜w_4 は各項の重みである．

交叉率 80%，突然変異率 10%，個体数 50 とし，各世代で適合度上位 1 個体をエリートとして次世代に残すように設定し，実験を行った．また，80 世代目で計算終了とした．総評価回数は 4.0×10^3 である．

比較のために SXX を用いた実験も行った．計算量 $O(n^2)$（n：順列中の要素数）でサブツアーを列挙するアルゴリズム[19] を用い，列挙されたサブツアーのペアからランダムに選ばれたペアを交換するようにした．

Pentium III 550 MHz，メモリ 128 M バイト，Windows NT4.0 の PC 上で C++を用いて実装したシステムで CCX，SXX とも 1 試行あたり約 70 秒を要した．

現実の JSSP で重要な要素は，評価の良い部分列と，その順列中の絶対位置であると考えられる．

そこで，まず適合度に総段取り替え時間のみを用いた．これは段取り替え時間の最適化に，作業順の部分列が重要な意味を持つことを調べるためである．式 (1) の重み w_1 〜 w_3 を 0 とし，第 4 項のみを適合度として用いた．CCX，SXX ともに作業順の部分列を親から子に伝える能力があるため，どちらの手法も総段取り替え時間を効果的に最小化できると考えられる．CCX と SXX それぞれの 100 試行中の総段取り替え時間の平均値（average）と標準偏差（stdev）を **表 2** の (a) に示す．予想したとおり，総段取り替え時間の最適化では CCX と SXX は同程度の最適化能力を持っている．

次に，適合度に総納期遅れ時間と倉庫コストを用いた．これは，最終工程の完了時刻と納期とのずれが小さいほど良い適合度となるため，順列中の各作業の位置が重要であることを調べるためである．式 (1) の重み w_1 と w_4 を 0 とし，第 2 項と第 3 項のみを適合度として用いた．CCX は作業順の部分列の絶対位置も次世代に伝える能力があるが，SXX にはないことから，CCX のほうが優れていると考えられる．100 試行の結果を表 2 の (b) に示す．CCX が SXX よりも良い結果となっている．

表 2 音響部品メーカのスケジューリング
Table 2 The result of the audio parts manufacturer.

	CCX		SXX	
	average	stdev	average	stdev
(a) Setup time	2216.9	5.7	2216.9	4.9
(b) Tardiness and stock	16.9	1.5	17.3	1.8
(c) All criteria	91.4	5.4	96.2	5.5

stdev: standard deviation

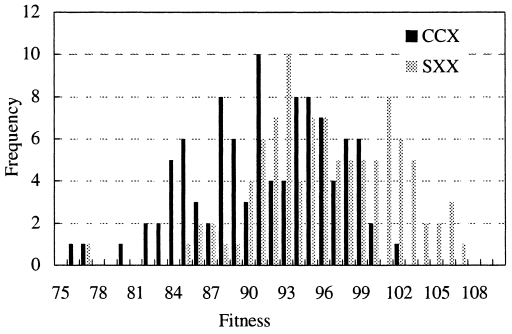

図 10 適合度（4 指標の重み付き和）の度数分布
Fig. 10 Frequency distribution of fitness.

続いて，式 (1) のすべての項を用いて，このメーカの目的である納期遅れ，倉庫コスト，段取り替え時間の削減を行った．100 試行の結果を表 2 の (c) に示す．また，**図 10** は，100 回試行を行ったときの最良個体の適合度（Fitness）の度数分布である．部分列と，その順列中の絶対位置の両方が重要となる現実の問題の複数指標最適化では，CCX の最適化能力が，SXX よりも優れている．

以上の実験結果から，現実の JSSP を最適化するには，「作業順の部分列」と「その順列中の位置」の両方が重要であり，これらの形質を次世代に伝える能力を持つ CCX は現実の JSSP を効果的に最適化できるといえる．

我々が提案する手法では 3.2 節で述べたように，CCX，疑似 CCX（Quasi-CCX），一点交叉（One-point crossover）の 3 種類の交叉が行われる．CCX が適用できない場合は疑似 CCX を適用し，さらに疑

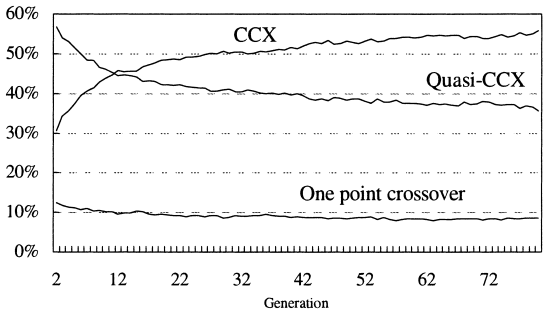

図 11 3 つの交叉の割合の変化
Fig. 11 Temporary change of the proportion of three kinds of crossover.

似 CCX も適用できない場合は一点交叉を適用する．**図 11** は，各世代で使用された交叉の種類の割合である．各値は 100 試行の平均である．はじめは CCX が 30%，疑似 CCX が 57%，一点交叉が 13%であったのが，世代交代とともに CCX の割合が増加し，80 世代目では CCX が 56%，疑似 CCX が 36%，一点交叉が 8%となっている．最適化が進むに従ってビルディングブロックとしての良い部分列が形成されていき，CCX の割合が増加している．

6. おわりに

本論文では，実用的な JSSP に適した新しい遺伝表現とコモンクラスタ交叉（CCX）を提案した．この手法では，染色体の遺伝子は作業順の入替え情報を表し，この情報を変換して作業の順序を得る．CCX は 2 つの親に共通な遺伝子のクラスタを交換して子を作る．この遺伝表現と CCX を使うことで，親が持つ作業順の部分列とその絶対位置を子孫に伝えることができる．

この手法を ft10x10 と ft20x5 の 2 つのベンチマーク問題に適用した．ft10x10 では SXX や GTX などの手法に劣るが，ft20x5 では SXX と同程度，GTX よりも優れた解を少ない評価回数で得られた．さらに，この手法を音響部品メーカのスケジューリングに適用し，納期遅れ，倉庫コスト，段取り替え時間の削減について評価を行った．この結果，本手法は従来の手法に比べて効果的にこれらの要素を最適化できた．このことから，本手法が実用的なジョブショップスケジューリング問題に非常に有効であるといえる．

参 考 文 献

1) Davis, L.: Job Shop Scheduling with Genetic Algorithms, *Proc. 1st Intl. Conference on Genetic Algorithms and their Applications*, pp.136–140 (1985).
2) Syswerda, G.: Schedule Optimization Using Genetic Algorithms, *Handbook of Genetic Algorithms*, Davis, L.(Ed.), Van Nostrand Reinhold, New York (1991).
3) Bierwirth, C.: A generalized permutation approach to job shop scheduling with genetic algorithms, *OR Spektrum*, Vol.17, pp.87–92 (1995).
4) Nakano, R. and Yamada, T.: Conventional Genetic Algorithm for Job Shop Problems, *Proc. 4th Intl. Conference on Genetic Algorithms*, pp.474–479 (1991).
5) Bruns, R.: Direct Chromosome Representation and Advanced Genetic Operators for Production Scheduling, *Proc. 5th Intl. Conference on Genetic Algorithms*, pp.352–359 (1993).
6) Grefenstette, J., Gopal, R., Rosmaita, B. and Gucht, D.V.: Genetic Algorithms for the Traveling Salesman Problem, *Proc. 1st Intl. Conference on Genetic Algorithms and their Applications*, pp.160–168 (1985).
7) Glodberg, D.E. and Lingle, J.R.: Alleles, Loci and Traveling Salesman Problem, *Proc. 1st Intl. Conference on Genetic Algorithms and their Applications*, pp.154–159 (1985).
8) Reeves, C.R.: A Genetic Algorithm for Flowshop Sequencing, *Computers and Operations Research*, Vol.22, No.1, pp.5–13 (1995).
9) Oliver, I.M., Smith, D.J. and Holland, J.R.C.: A Study of Permutation Crossover Operators on the Traveling Salesman Problem, *Proc. 2nd Intl. Conference on Genetic Algorithms*, pp.224–230 (1987).
10) Whitley, D., Starkweather, T. and Shaner, D.: The Traveling Salesman and Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination, *Handbook of Genetic Algorithms*, Davis, L.(Ed.), Van Nostrand Reinhold, New York (1991).
11) Starkweather, T., McDaniel, S., Mathias, K. and Whitley, D.: A Comparison of Genetic Sequencing Operators, *Proc. 4th Intl. Conference on Genetic Algorithms*, pp.69–76 (1991).
12) 小野 功，佐藤 浩，小林重信：サブシーケンス交換交叉と GT 法に基づくジョブショップスケジューリングの進化的解法，電気学会論文誌，Vol.117-C, No.7, pp.888–895 (1997).
13) Baker, J.E.: Adaptive Selection Methods for Genetic Algorithms, *Proc. 1st Intl. Conference on Genetic Algorithms and their Applications*, pp.101–111 (1985).
14) Liepins, G.E. and Potter, W.D.: A Genetic Algorithm Approach to Multiple-Fault Diag-

nosis, *Handbook of Genetic Algorithms*, Davis, L.(Ed.), Van Nostrand Reinhold, New York (1991).
15) Fisher, H. and Thompson, G.L.: Probabilistic learning combinations of local job-shop scheduling rules, *Industrial Scheduling*, Muth, J. and Thompson, G.L.(Eds.), Prentice Hall, Englewood Cliffs, New Jersey (1963).
16) Giffler, B. and Thompson, G.L.: Algorithm for Solving Production Schedule Problems, *Operations Research*, Vol.8, pp.487–503 (1960).
17) Yamada, T. and Nakano, R.: A Genetic Algorithm Applicable to Large Scale Job Shop Problems, *Parallel Problem Solving from Nature*, Vol.2, pp.281–290 (1992).
18) 辻村泰寛，玄　光男，真船雄一郎：GA によるジョブショップ・スケジューリングにおける評価関数とスケジューリング構造の関係，信学技報（AI99-13），Vol.99, No.96, pp.17–24 (1999).
19) 柳浦睦憲，永持　仁，茨木俊秀：サブツアー交換交叉に対する 2 つのコメント，信学技報（COMP94-18），Vol.94, No.88, pp.1–10 (1994).

(平成 12 年 7 月 21 日受付)
(平成 13 年 6 月 19 日採録)

手塚　大

1994 年大阪大学大学院基礎工学研究科物理系専攻博士前期課程修了．同年，日立東北ソフトウェア（株）入社．生産スケジューリングソフトウェアの研究，開発に従事．

樋地　正浩（正会員）

1986 年山形大学理学部物理学科卒業．同年，日立東北ソフトウェア（株）入社．1997 年東北大学大学院情報科学研究科システム情報科学専攻博士課程修了．博士（情報科学）．自律分散協調システム，CSCW の研究，開発に従事．電子情報通信学会，ソフトウェア科学会，認知科学会，ACM，IEEE/CS 各会員．