人　工　知　能　100－1
（1995．5．24）

事例を用いた探索効率化学習

宮下和雄

電子技術総合研究所

筆者らは悪構造問題における最適化と知識獲得を統合化したシステムとしてCABINS(CAse-Based Intelligent Scheduler)を開発した。CABINSでは、ユーザによる最適化のための解修正過程が、修正対象の文脈情報と共に事例として獲得される。CABINSによる自動的な反復修正過程において、蓄えられた事例は修正方法の決定と修正結果の評価に用いられる。本論文では強力なドメインモデルを持たない問題に関して、過去の失敗事例を用いて最適化プロセスを効率化する手法を提案し、その有効性を典型的な悪構造問題であるジョブショップスケジューリング問題を用いた実験により示す。

Case-Based Speed-Up Learning

Kazuo Miyashita

Electrotechnical Laboratory
1-1-4 Umezono, Tsukuba, Ibaraki 305, Japan

We have developed an integrated framework of iterative revision and knowledge acquisition for ill-structured optimization problems, and implemented it in the CABINS system. In CABINS, situation-dependent user's decisions that guide repair process are captured in cases together with contextual information of the problems. During iterative revision in CABINS, cases are exploited for both selection of the appropriate repair action and evaluation of repair results. In this paper, we experimentally demonstrate that in a problem where no strong domain model is available, past repair failure experiences can be exploited to improve the efficiency of rather intractable iterative optimization process in multiple ways such as (1) for validating effectiveness of selected repair actions and (2) for giving-up further repair of too difficult problems. The experiments in this paper were performed in the context of job-shop scheduling problems, which are well-known ill-strucured problems.

1 序論

[8] では、筆者らは事例に基づく反復最適化システム CABINS が、悪構造な問題領域においてユーザの持つ文脈依存の選好条件を獲得し、解品質の改善に有効に活用できることを示した。しかし、ジョブショップスケジューリング問題をはじめとする多くの悪構造問題において、高品質の解を生成するには膨大な空間の探索を必要とする。そうした問題を解決するには、問題空間を効率的に探索するために有効な制御知識を持つことが重要である。本論文では、筆者らは CABINS が過去の失敗経験を (1) 修正戦略の承認、(2) 現在の修正対象に対する修正中断などの決定に用いることにより、問題解決効率を改善することができることを示す。

過去の悪構造問題における効率改善学習の研究 [10] では、説明に基づく学習を拡張した手法が NASA におけるスケジューリング問題の制御知識獲得に用いられた。そのシステムでは、問題状況に応じてスケジューリングヒューリスティクスを動的にスイッチングするための知識が獲得されたが、そこで選択対象とされたヒューリスティクスはわずか 2 種類で、一般的な最適化問題に対しての有効性を論じるには不適切である。それに対し、CABINS が学習できる概念は、特定の問題状況における個々のヒューリスティクスの適切性や特定の問題に対する解決困難性など、より一般的で有用なものである。

近年、類推による推論を用いて効率化学習を行なう研究がいくつかなされている。(例えば、PRODIGY [9]、EUREKA [4]、DÆDALUS [5]) これらのシステムは目標や前提条件、オペレータによりモデル化を行なえる問題に対して、目的手段解析によりプランを作成、保存し、そのプランの一部を新たな問題解決に再利用することで問題解決の効率化を図っている。しかし、CABINS が対象としている問題には、そのように強固な領域モデルは存在しないため、それらの手法は適用できない。領域モデルの不足を補い、問題解決効率を改善するために CABINS は積極的に過去の失敗事例を用いている。[1]

以下、本論文では 2 章で CABINS における事例に基づく修正手法を概説し、3 章でジョブショップスケジューリング問題を用いた実験手法について説明する。続く 4 章では CABINS における 3 種類の異なった失敗事例活用手法を述べ、効率改善に対する各々の手法の有効性を実験的に確認する。最後に 5 章では本研究の成果の要約を行なう。

2 CABINS: 事例に基づく修正手法

CABINS(図 1) は以下の 3 モジュールにより構成されている：(1) 初期解生成器、(2) インタラクティブな修正（事例獲得）モジュール、(3) 自動修正（事例再利用）モジュール。初期解の生成方法として CABINS は

[1] EUREKA と DÆDALUS は成功事例のみから学習を行なう。PRODIGY は過去の失敗事由が現在の問題と完全に一致した場合にのみ、探索空間の枝刈りを行なう。

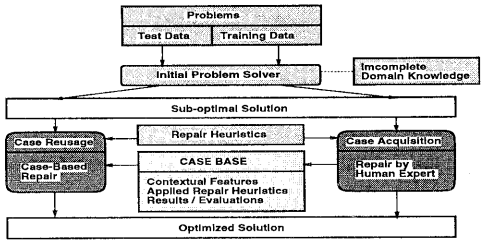

図 1: CABINS アーキテクチャ

幾つかのオプションを持つが、領域知識やユーザの選好知識が不完全なため、一般的にどの手法を用いても最適解を生成できない。それらの知識の不足を補い、より良い解を生成するため、CABINS は生成された初期解をユーザと対話的に修正する過程を事例として記録することにより以下の情報を獲得する：(1) 修正箇所：ユーザは与えられた解における最も致命的な問題点を指摘する、(2) 修正方法：ユーザは問題点を解決するため最も有効と思われるヒューリスティクスを選択する、(3) 結果評価：ユーザは修正結果の良否を大域的に判断する。十分な数の事例が集まると、CABINS は事例ベース推論を用いて、ユーザを介さずに初期解をユーザの満足いく程度の品質にまで効率的に改善することができる。

2.1 CABINS によるスケジュール修正

本論文では、良く知られた NP 困難な問題であるジョブショップスケジューリング問題を用いて CABINS の性能評価を行なう。ジョブショップスケジューリング問題はジョブの集合 $O = \{O_1, \ldots, O_n\}$ を有限容量の物理的資源の集合 $RES = \{R_1, \ldots, R_m\}$ に割り付ける問題である．各々のジョブ O_l は相互の順序関係が決められた幾つかのアクティビティ $A^l = \{A^l_1, \ldots, A^l_{n_l}\}$ から構成されている。また、各々のジョブ O_l には投入時期 rd_l と納期 dd_l が決められている。個々のアクティビティ A^l_i は固定の加工時間 du^l_i と可変の開始時間 st^l_i が与えられる。その開始時間はアクティビティが属すジョブの投入時期と納期に拘束される。

CABINS は修正に基づく最適化における異なるレベルの意思決定に対応して 3 クラスの修正行為 (即ち、ゴール設定および戦略／戦術適用) を備えている。図 2は以下のステップから成る CABINS の事例に基づくスケジュール修正プロセスを図示したものである。

1. スケジュールを評価する各指標値が計算され、CBR を用いて過去のユーザの選好に基づき最も致命的な欠陥をもつ指標（即ち、修正ゴール）を決定する。
2. もし、修正ゴールが "give-up" なら、CABINS は修正を終了する。
3. 選ばれた欠陥を持つジョブを欠陥値の降順にソートし、その順番に従って個々のジョブを修正する。

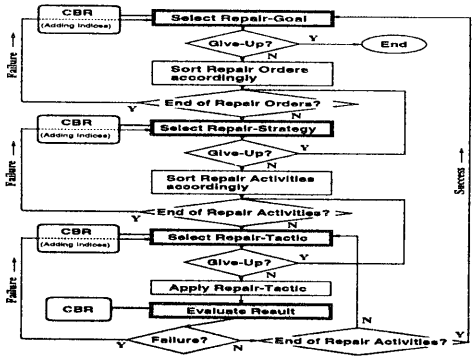

図 2: CABINS による修正に基づく最適化手順

4. 修正対象のジョブに対して、CBR を用いて修正戦略を決定する。

5. もし、修正戦略が "give-up" なら、CABINS は現在のジョブの修正を中断し、他のジョブの修正に移行する。

6. 修正対象のジョブに属するアクティビティを修正戦略に基づきソートする。これはアクティビティの移動に伴う制約伝播の程度をできるだけ小規模に抑えるためである。アクティビティはこの順序に従って個々に修正される。

7. CBR を用いて修正対象のアクティビティに対して修正戦術を決定する。

8. もし、修正戦術が "give-up" なら、CABINS は現在のアクティビティの修正を中断し、他のアクティビティの修正に移行する。

9. 選択された修正戦術は以下のように適用される：(1) 戦術に基づき対象アクティビティを移動、(2) それによってもたらされた制約違反を解消。

10. CABINS は修正結果を CBR を用いて評価する。

11. 評価に応じて、CABINS は以下のどちらかの処理に進む。

    (a) 修正結果が成功と判定された場合、次のアクティビティの修正に進む。1 つのジョブの修正が終ると、次のジョブの修正に移行するのではなく、修正ゴールの再設定を行なう。これは、一連の修正作業によりスケジュールの性質が大きく変わった可能性に対応するためである。

    (b) 修正結果が失敗と判定された場合、CABINS はこの失敗情報を索引として追加し、CBR を用いて新たな修正戦術を選択する。これにより、CABINS は過去に同様の失敗をしながら最終的に修正に成功した類似事例を選択することができる。ここでは筆者らは、同一の修正により同様の失敗を生じた事実は過去の事例と現在の問題の因果的類似性を示しているとの仮定に立っている。

CABINS には現在 3 つの修正ゴール、5 つの修正戦略、11 の修正戦術が実装されているが、ユーザの要望に応じて容易に追加、変更することが可能である。

3 CABINS の評価

筆者らは解品質や探索効率に関して、CABINS と他の最適化手法との比較を行なった。本論文中の実験では、納期遅れと中間在庫の和を仮想的なユーザの最適化指標として用いた。更に、この指標を最小化するように欲張り戦略を用いて試行錯誤で解修正を行なう山登り最適化プログラム (GHO) を実装した。GHO は 1 つのアクティビティの修正において、各修正アクションによる修正効果を見積り、最も効果の大きいと判定された修正アクションから順に、修正が成功するか、適用可能なアクションが無くなるまで修正を続ける。一回の修正毎に、GHO はコスト関数に基づき修正効果を評価し、修正の成否を判定する。GHO は CABINS の事例生成のために用いられ、GHO のよる修正アクションの選択や修正結果評価は事例ベースに蓄えられた。GHO は正確に与えらたコスト関数を用いるが、CABINS は GHO により生成された事例の集合に基づいて外延的にのみ修正結果の良否を評価できる。また、GHO は CABINS の性能を評価するためのベースラインとしても用いられた。更に、GHO は局所最適解に陥る可能性が高いため、確率的な最適化手法であるシミュレーティッドアニーリング手法 (SA) [3] と CABINS の比較も行なった。

筆者らはボトルネックマシンの数やジョブの納期や加工時間をパラメータとした 6 種類のジョブショップスケジューリング問題に対し、各 10 題ずつ問題をランダムに生成することにより、計 60 問の問題を用いてベンチマークテストを行なった。用いられた各々の問題は 5 つの機械と 5 アクティビティから成るジョブを 10 ずつ持ち、各ジョブのアクティビティは前後の工程関係が決められており、ボトルネックマシンへの負荷が一定以上に大きくなるように設定されている。

このようにパラメータを制御することにより、ある近傍内の問題群を生成し、それらの問題を用いてスケジューリング手法を評価することは、AI や OR の研究で良く行なわれている。これらの問題はランダムに生成はされているが、共通な特徴 (例えば、機械やジョブ、アクティビティの数) も備えている。こうした事情は、常に全く同一ではないものの多くのルーティン的な仕事が行なわれる現実の工場などとも共通したな状況である。CBR は、このような問題構造の規則性を新たな問題の解決のために用いることができる。

筆者らは CABINS の性能を評価するのにクロスバリデーション法を用いた。即ち、問題集合を 2 等分し、一方の問題を GHO で解くことによって得られた事例を用いて、他方の問題集合に属する問題に対する CABINS

による修正を行ない、その結果を評価した。本論文では筆者らは約12000個の事例を用いて実験を行なった。

3.1 実験結果

表 1: CABINS、GHO及びSAの性能比較

	Solution Cost	Tactic Applications
CABINS	698.8	2206.2
GHO	752.4	1229.8
SA (average)	812.6	12568.8
SA (best)	596.3	125688.2

表1にCABINS、GHO及びSAの性能評価結果を示す。表中のデータは60問の問題を解いた結果の平均値である。SAに関しては確率的な探索手法であるので、各々の問題を10回ずつ解き、その平均値と最高値を記している。表1において、解品質を示す指標として解コスト（納期遅れと中間在庫量の和）を用い、また問題解決効率を示す指標として修正アクションの適用回数を用いた。修正アクションの適用は制約伝播や修正効果の計算を含むため、CABINSによるスケジュール修正プロセス中で最も計算負荷が大きい。したがって、修正アクションの適用回数を削減することにより、CABINSの繰り返し修正プロセスの効率を改善することができる。

得られた結果から、CABINSはGHOと同レベルの品質の解を得ることができるものの、そのためにGHOよりも80%程度多くの修正アクションの適用を行なったことがわかる。このことはCABINSが解品質の改善に有効でない修正アクションの選択、実行を繰り返し行なっていることを意味している。SAは確率的な探索を行なうため、膨大な探索を行なった結果、非常に高品質な解を生成することができる可能性がある。しかしながら、平均的な結果で比較を行なうと、SAはGHOやCABINSに比べて、得られる解品質に対する問題解決効率が劣っている。

ジョブショップスケジューリングのような悪構造な問題においては、専門家でさえも解の改善に有効な修正アクションを正確に予測するために必要な問題の特徴量に関する十分な知識は持っていない。しかも、スケジュールにおける制約は密接に関連し合っているため、問題を完全に表現するために必要な問題特徴量の数は非常に大きい。その結果、現在のCABINSの事例表現で用いられている特徴量の数は明らかに不十分である。しかし、事例に含まれる特徴量の数が多くなると、概念を正しく帰納学習するのに必要な訓練事例の数が飛躍的に増加する(dimensionality problem)。したがって、適当な数の問題特徴量と訓練事例を用いた場合、事例ベース学習を含むどんな帰納的学習手法も誤った予測をすることは避けられない。このことが表1中のCABINSの非効率性の原因と考えられる。

4 CABINSにおける効率改善

事例を表現する特徴量の不足を補う目的で、筆者らは事例における失敗経験を予測精度の改善（即ち、より関連する事例を事例ベースから検索する）に利用した。我々は、過去の失敗事例は以下の2点でCABINSの効率改善に役立つと考えた：(1)過去に経験した失敗の繰り返しを避ける、(2)困難過ぎる問題を解決しようとする無駄な努力を避ける。言い替えると、CABINSは失敗事例より以上の能力を持つ制御知識を学習することができると考えた。

以上の仮説の正しさを確かめるために、以下の3種類の修正手法をCABINSに実装し実験を行なった:(1) Validated repair、(2) Interruptive repair、(3) Hybrid repair。これらの修正手法に対して、3章と同一の問題、コスト関数、事例ベースを用いて実験を行なった。

4.1 修正における失敗事例の活用

Validated repair手法では修正アクションは現在の問題の修正に有効であることが承認された後に初めて実際に適用される。このような承認テクニックは、以前より事例ベース推論システムで広く用いられている（例えば、[6, 2]）。CHEF[2]ではMODIFIERモジュールがドメインの因果モデルを用いて検索されたプランの検査、変更を行なっている。その結果、修正されたプランが実行中に失敗する可能性は小さい。CASCADEやCOAST[6, 7]といったヘルプデスクシステムでは、ドメインの承認モデルを用いて現在の問題に密接に関連しそうな事例のみを事例ベースから検索する。ドメインの承認モデルには、診断のための個々のテストに関する知識や、異なるテスト間の相互関係に関する知識が含まれる。充実した承認モデルを用いることにより、面倒なテストの適用に伴う探索コストを減らすことができ、広大なテスト空間を効率的に探索することができる。

以上のシステムはドメインに対する深い理解に基づいて承認を行なっている。しかし、ジョブショップスケジューリングのように悪構造な問題においては因果モデルや承認モデルは容易に構築できない。例えば、ジョブショップスケジューリング問題においては、非線形な目的関数や密な制約相互作用のため、局所的な最適化操作が大域的な最適性にどう影響を及ぼすかを予測するのは困難であり、因果モデルや承認モデルを作るためのドメイン構造の解析が非常に難しい。したがって、CABINSでは検索された事例の承認をそのようなモデルなしに行なう。モデルが利用できない替わりとして、CABINSは選択された修正アクションの正当性を評価するために過去の失敗経験を利用する。

図3にCABINSの承認プロセスの概要を記す。修正アクションの選択に当たり、CABINSはまず過去における成功した修正エピソードである成功事例から、現在の問題と類似した候補事例を検索する。つぎに、過去の失敗経験を蓄えている失敗事例から、選択された候補事例で用いられたのと同一の修正アクションを用いた類似

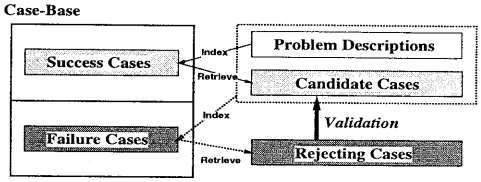

図 3: CABINS における承認プロセス

事例である拒絶事例を検索する。この時、拒絶事例の信頼性が低い場合、すなわち拒絶事例と現在の問題との類似性が低い時は、候補事例の有効性（修正失敗の可能性の少なさ）が承認される。

修正中断を許すことにより、CABINS は非常に困難な問題の修正に無駄な労力を費やす前に、計算資源を問題の他のもっと容易に解決できそうな領域に振り替えることができる。CABINS はそれ以上の修正努力が時間の無駄に終りそうな時に、その問題に対する修正を中止する。それ以上の修正を中止するか否かの判断には、CABINS では失敗事例が用いられる。すなわち、過去に現在の状況と類似した失敗事例が多く存在するときは、CABINS は現在の問題の修正を中断し、他の問題の修正に移る。

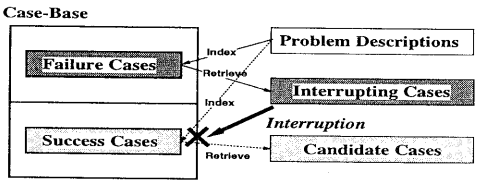

図 4: CABINS における中断プロセス

図 4に CABINS の中断プロセスの概要を記す。中断プロセスにおいては、まず現在の問題状況と合致する失敗事例が中断事例として検索される。もし、この中断事例の信頼性が高い（すなわち、現在の問題との類似性が高い）時は、現在の問題に対する修正アクションを選択するために過去の成功事例から候補事例を検索するという処理は実行されず、CABINS は他の修正対象の修正を実行する。

以上に述べた承認プロセスと中断プロセスにおける失敗事例の活用方法は、互いに独立で、両立できると考えられる。すなわち、承認プロセスと中断プロセスは、修正アクション選定プロセスにおける異なった局面において、失敗事例を探索速度の向上のために用いている。したがって、Validated repair 手法と Interruptive repair 手法を単純に組み合わせることにより、Hybrid repair 手法を実現することができる。

4.2 実験結果

表 2に、オリジナルの（すなわち、失敗事例を用いない）CABINS、GHO および 3 種類の失敗事例を活用した CABINS の性能比較を行なった結果を記す。

表 2: CABINS における失敗事例活用による性能比較

	Solution Cost	Tactic Applications
CABINS	698.8	2206.2
GHO	752.4	1229.8
Validated Repair	751.1	920.3
Interruptive Repair	724.3	1186.8
Hybrid Repair	753.4	776.6

表中の結果によると、Validated repair 手法を用いた CABINS はオリジナルの CABINS に比べると、スケジュール品質を著しく損なうことなしに、効率を約 58% も改善していることがわかる。さらに、Validated repair 手法を用いた CABINS は GHO よりもずっと効率的であることがわかる。Validate repair 手法を用いた CABINS の GHO に対する優位性は Etzioni の仮説検定手法 [1] により検定され、5% の有意度で Validated repair 手法を用いた CABINS が GHO 手法よりも効率的であることが検定された。

さらに、表 2の結果によれば、Interruptive repair 手法を用いた CABINS はオリジナルの CABINS と比較して、高いスケジュール品質を維持しながら、約 46% も効率を改善していることがわかる。GHO との効率の比較においては、Interruptive repair 手法を用いた CABINS は検定による有意さは認められなかった。しかしながら、オリジナルの CABINS との比較においては、Interruptive repair 手法の有効性は明らかである。

最後に、表 2中の結果によると Hybrid repair 手法を用いた CABINS はオリジナルの CABINS に比べて、スケジュール品質を大きく損なうことなく約 65% もその効率を改善していることがわかる。さらに、表中の結果によると、Hybrid repair 手法を用いた CABINS は、GHO よりも効率的に高品質なスケジュールを生成することができることがわかる。このことは先の Etzioni の仮説検定手法によっても確認された。

Hybrid repair 手法において、Validated repair と Interruptive repair という 2 つの修正手法を組み合わせたことによる相乗効果を分析するため、Validated repair 手法と Interruptive repair 手法による修正戦術適用回数が Hybrid repair 手法による修正戦術適用回数と比較された。仮説検定の結果、Hybrid repair 手法は Validated repair 手法や Interruptive repair 手法よりも効率的であることが認められた。したがって、Validated repair 手法と Interruptive repair 手法は互いに独立で両立でき、修正プロセスの効率化のための相乗効果が生み出すことが確認された。

4.3 考察

4.2章における実験の結果、CABINS において失敗事例の活用は修正戦術適用回数の削減に有効であるこ

とが示された。しかしながら、Validated repair、Interruptive repair、Hybrid repairによる戦術適用回数の削減は、オリジナルのCABINSに比べてより複雑で計算時間のかかる失敗事例を活用した事例検索手法に、その成功を負っている。したがって、失敗事例を活用するこれらの手法の、オリジナルなCABINSに対する効率面での優位性は、実際のスケジュール修正に要したCPU時間でも評価する必要がある。

表 3: 種々の修正手法を用いたCABINSによる修正時間の比較

	CPU time (sec)
Validated Repair	435.9
Interruptive Repair	528.1
Hybrid Repair	424.1
CABINS	551.1

表3に種々の修正手法を用いたCABINSによるスケジュール修正に要した時間の平均値を記す。この表で注意すべき点は、現在のCABINSの実装では事例メモリが単純な線形リストとして実現されているため、表中のCPU時間の大部分は実際のスケジュール修正アクションの実行ではなく、事例の検索に用いられていると考えられる点である。

この表によると、上記の3手法を用いたCABINSはオリジナルなCABINSに比べて、CPU時間に関しても削減していることがわかる。しかし、それらの3手法によるCPU時間の削減の割合は、失敗事例を用いたより複雑な事例検索プロセスが戦術適用回数削減の効果をCPU時間に関して相殺してしまうため、戦術適用回数削減の割合に比べると小さなものとなっている。この結果は、本論文の実験では、失敗事例の数は成功事例に比べてはるかに多いため、失敗事例を活用した事例検索は成功事例のみを用いた事例検索に比して、検索に伴う計算時間をより多く必要とすることからも説明される。

5 結論

筆者らは、ジョブショップスケジューリングのような強力な領域モデルを持たないドメインにおける計画修正プロセスを制御するために、事例を獲得し再利用する枠組について述べ、その有効性を実験により評価した。筆者らはそのようなドメインにおいて効率を改善するために、過去の失敗経験を利用する様々な手法を検討した。ジョブショップスケジューリング問題における実験の結果、過去の失敗事例を(1)修正アクションの承認や(2)困難過ぎる問題からの修正視点のシフトに用いることにより、修正効率の著しい改善が実現されることが確認された。筆者らは、CBRにおけるこの失敗事例の活用手法は、問題解決器に対して、強力な領域モデルを持たないドメインにおける制御知識を獲得し、問題解決効率を改善するための領域に依存しない手法であると考える。

参考文献

[1] Oren Etzioni and Ruth Etzioni. Statistical methods for analyzing speedup learning. *Machine Learning*, 14(3):333–347, 1994.

[2] Kristian J. Hammond. *Case-Based Planning : Viewing Planning as a Memory Task.* Academic Press, New York, NY, 1989.

[3] D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon. Optimization by simulated annealing: An experimental evaluation, part I (graph partioning). *Operations Research*, 37(6):865–892, 1989.

[4] Randolph Jones. Problem solving via analogical retrieval and analogical search control. In Alan L. Meyrowitz and Susan Chipman, editors, *Foundations of Knowledge Acquisition: Machine Learning.* Kluwer Academic, Boston, MA, 1993.

[5] Pat Langley and John A. Allen. A unified framework for planning and learning. In Steven Minton, editor, *Machine Learning Methods for Planning.* Morgan Kaufmann, San Mateo, CA, 1993.

[6] E. Simoudis and J.S. Miller. Validated retrieval in case-based reasoning. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 310–315. AAAI, 1990.

[7] E. Simoudis and J.S. Miller. The application of CBR to help desk applications. In *Proceedings of the Case-Based Reasoning Workshop*, pages 25–36. DARPA, 1991.

[8] Katia Sycara and Kazuo Miyashita. Case-based acquisition of user preferences for solution improvement in ill-structured domains. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 44–49, Seattle, WA, 1994. AAAI.

[9] Manuela M. Veloso. *Learning by Analogical Reasoning in General Problem Solving.* PhD thesis, School of Computer Science, Carnegie Mellon University, 1992.

[10] M. Zweben, E. Davis, D. Brian, E. Drascher, M. Deale, and M. Eskey. Learning to improve constraint-based scheduling. *Artificial Intelligence*, 58(1-3):271–296, 1992.