人　工　知　能 76－1
（1991．5．22）

Incremental Goal Seeking モデルに基づくエキスパートシステムの提案

江藤 博明　　丸山 圭一
日本アイ・ビー・エム (株) 東京基礎研究所

本論文では，予算計画のような線形の制約式を満たす必要があり，評価が複数かつ形式化の困難な計画問題を，計画者とシステムが協調的に解いていくシステムについて述べる．

このシステムは計画問題の制約式を満たす計画エンジンを有している．そのため計画者は，制約式を満たすことを考えることなく，計画の候補を得られる．さらにこのシステムは計画者の評価に基づいた計画変更の指示を受け取るユーザインタフェースを設けている．その指示をシステムは制約として取り扱い，計画エンジンを用いてそれを反映した計画の候補を提示する．この操作を繰り返すことによって計画者は自身の評価に基づいた計画に到達できる．

A proposal for an expert system based on the incremental goal seeking model

Hiroaki Etoh　　Keiichi Maruyama
IBM Research, Tokyo Research Laboratory

This paper describes a cooperative system that helps a planner to solve a planning problem in which linear constraints must be satisfied and its evaluation criteria are plural and difficult to difine.

The system has a planning engine that satifies constraints of a problem. Therefore a planner gets a plan candidate without considering to satisfy constraints. Moreover it has a user interface that receives directions based on his evaluation. It translates them as constraints and proposes a plan candidate that reflects them by using the planning engine. He reaches a plan based on his evaluation criteria by repeating the operation.

1 はじめに

予算計画等のビジネスプランニング業務は一般に難しく，プラン作成に時間が掛かる．その上，時間的に早く作成することを求められることがビジネス上多く，機械化が望まれる分野である．

予算計画の難しさは第一に計画には満たすべき制約があり，それを満たした計画案を得ることに多大な労力を必要としてしまうことにある．第二の難しさは計画の評価基準がいくつもある上，それらを形式化することが困難なことである．計画の評価を下すには専門家としての経験と能力を必要とする．このことが機械化を難しくしている．

本論文では，予算計画とは何かについて定義（節２）し，それを解決するためには何が必要なのかを検討（節３）する．

ここで提案するシステムはルールまたは専門家による評価値の指示から計画案を作成していく協調型システム（節５，節６）である．

2 予算計画

この節では，予算計画の特徴について例を示しながら述べ，予算計画の専門家についての特徴を仮定する．

図１は典型的な資金の運用計画表である．各カラムは投資項目，投資する額，全体に占めるパーセンテージ，投資に対する利益率および投資から得られる利益である．最下段はその合計値であり，運用可能な資産の合計，平均の利益率および利益の合計が示されている．

	A	B	C	D	E
1	項目	予算額	割合	利益率	利益
2	現金	149659	10.3	2.16	3232.6
3	有価証券	236008	16.2	7.48	17653.4
4	貸し出し金	1045046	71.9	5.81	60717.2
5	不動産	23404	1.6	0.00	0.0
6	資産合計	1454117	100	5.61	81603.2

図 1: 予算計画の例

1. 予算計画の作業は，３で述べる制約を満たすような各数値要素の値の集合から，４で述べる目標を達成する数値要素の値を決定することである．この例では運用計画の総予算額（実は調達された資金額）を，目標を達成するように，各資産へと配分することである．

2. 予算計画における数値は，連続的な値を持つ．例えば，現金の予算額は，０から資産合計まで連続的な値を割り当てることができる．

3. 予算計画における制約とは，各数値要素間に成立すべき関係で表現されるもので，等式制約と不等式制約に分類される．議論の都合上，等式制約および不等式制約は線形の制約式で表されるものと仮定する（節７で，曖昧な制約式について考察する）．例えば，運用計画における制約は，

   (a) 資産合計は調達金額と同額でなければならない．つまり資産合計は固定である．

   (b) 予算額の合計は資産合計に等しい．

   (c) 利益の合計は総利益に等しい．

   (d) 有価証券は全体の１７％以下．

   (e) 貸し出し金は 1000000 以上，1050000 以下．

   のように，等式制約と不等式制約によって記述される．また，線形の制約にするために，各資産における利益率は定数値であると仮定する．

4. 予算計画における評価は，利益を最大にするといった，一つの評価であることは稀で，一般には利益とリスクのような，複数のトレードオフの関係にある評価からなる．それぞれの評価は線形の評価関数によって表わされる．ここまでの定義は，多目的線形計画 [2] の定義そのままである．予算計画に特有な特徴は，評価において以下の性質を有するところである．

   (a) 評価関数のすべてをあらかじめ，定義しておくことは難しい．というのは，専門家自身がすべての評価関数を認識しているとは限らない．運用計画では，リスクに関して多くの観点から評価することができ，専門家はその時々によって，評価関数を適用している．

   (b) 複数の評価関数に関してあらかじめ，順序付けすることは難しい．

(c) 周りの環境によって評価関数が変化することがある．例えば，公定歩合に応じてリスクの評価方法が変化するなどである．

評価に関して，以上のような曖昧さがあるにもかかわらず，専門家は予算計画作業を遂行している．ここで，専門家の特徴を一つ仮定しよう．

1. 上記の予算計画において，異なる二つの計画が与えられた時に，専門家は必ず，相対的な評価（等しい場合を含む）を下すことができる．

3 目標探索用の協調型システム

節2で述べたように，予算計画における評価をあらかじめ定義することが，現実的に不可能である．このような場合に，専門家が直接計画システムの動作へと介入できるようにして，システムと協調的に計画することが有効である．森下ら [5] が述べているように，協調型システムを設計する際のポイントは，専門家から求められる要件から検討を始めることである．この節では，専門家からのシステムの要件を決めた後に，計画エンジン，ルールおよびユーザインターフェースの要件を洗い出す．

3.1 専門家から求められるシステムの要件

節2で述べたように，予算計画において，専門家の判断が必要とされるのは，計画を評価する作業である．また，専門家は一つの評価値に着目した場合，評価値をどのように調整したら，より良い計画になるのか判っている．そこで専門家が求めるシステムの要件は，次のものを考えた．

要件1 専門家は制約について関知しない．システムが制約を満たす．

要件2 評価値の操作で新しい計画案を得る．

二つの要件を満たすシステムの実現可能性を検討している時に，我々は次のアイデアを得た．

- 専門家がこのシステムを使用することで，ある制約を満たしている計画案から，より良い計画を見つけられ，この手続きを繰り返すことで，最終的な予算計画に到達できる．

このアイデアについては，考察（節7）で検討するが，ここでは，このアイデアが正しいとして，システム要件の検討を続ける．

専門家は計画案を検討することで，その計画案に絶対評価を与えられるものであろうか？予算計画では，絶対評価は存在せず，現在の計画案と新しい計画案との間の相対評価を，専門家自身が与えることで，現在の計画案の良し悪しを判断するしかない．そこで現在の計画案を中心にして数多くの試行錯誤が必須の作業となる．そこで，次を要件として加える．

要件3 現在の計画案を中心に，複数の計画案を得られる．

要件4 簡単な操作で，新しい計画案が得られる．

評価値から計画を作成するときに，同じ評価値を持った計画が無数に存在している．作成される計画がシステム任せであると，専門家にとって思いがけない計画が作成されるかもしれない．最後の要件として，次を加える．

要件5 専門家は，作成される計画案の傾向を制御できる．

3.2 システム・コンポーネントの要件

前節で，挙げたシステム要件を，どのシステム・コンポーネントで処理するかによって，システム・コンポーネントの要件は変わる．我々は，システム要件1，2，3および5を計画エンジンで実現することにした．制約を満たすことや指定された評価値の計画を作成する作業は，このシステムで最も重い処理であり，また，要件の4で示した協調型システムとして，専門家との対話性を満たすため，以下のような要件を加える．

エンジンの要件 対話性を実現する処理速度．

ルールの役割は，専門家が計画エンジンを利用したことから得られるルーチンワーク的な手続きを処理することである．ルールは条件によって起動される手続きを記述する．そのルールとして状況によって変化する評価関数の処理が記述できる．

ルールの要件 条件によって起動される計画エンジンの処理を記述できる．

ユーザーインターフェースの役割は，システム要件2，4および5を計画エンジンに伝える事である．

4　従来のアプローチ

従来，節２で述べられた計画問題は，ＯＲの分野で多目的線形計画法や線形目標計画法の問題として適用されてきた．問題を形式化する際に前提となるのは，すべての制約とすべての目標関数をあらかじめ定式化することである．この条件から予算計画をＯＲで扱う事は難しい．

また，形式化が困難であっても，専門家にインタビューすることで，一応の形式化を行なうことは可能である．しかしながら，その問題を解いた結果を専門家に見せることで，形式化した以外の条件が現れたりする．この新たな条件のために，ＯＲ手法では解法が変わることがある．この解法を変更する作業は，ＯＲ的な手法では困難である．

次に，ＡＩ的な手法による予算計画の適用について述べる．予算計画問題を制約充足問題と捉えることで，制約充足機構を持っているシステム CLP[7]，Inspire[6] などで処理できるであろう．しかしながら，予算計画における目標を制約として処理しなければならないため，その解釈の仕方が非常に難しい．例えば，予算計画では利益の増大は目標の一つであるが，制約として捉えるために，利益として適当な額を達成しなければならないと考える．この額の設定によっては，計画が存在しない事があるし，計画が存在しても，目標が低過ぎることもある．このように制約充足の面からでは，予算計画を処理する事は難しい．

5　計画手順の概略

節２で述べた予算計画の例に対して，次の状況を考え，本システムで計画する際の流れを示す．「年初の予算計画は図１であったが，金利の変動により貸し出し金の利益率が５．８１から５．３に下がり，利益の合計が落ちてしまった．この時，資産合計を年初の目標を達成するために計画を作り直さなければならない．」

まず図１のスプレッドシートに対して，節２で述べた５つの制約を入力する．次にスプレッドシートの貸し出し金の利益率を５．３％に設定する．ここで調整したい数値を選択してから，調整用グラフを表示する（図２）．

我々の計画方法は調整項目の値をグラフ表現し，専門家は，このグラフ表現したもので値の調節を行なうものである．値の変更はグラフをマウスでドラッグすることにより行なう．

予算計画問題のグラフ機構による，実際の操作は以下の通りである．

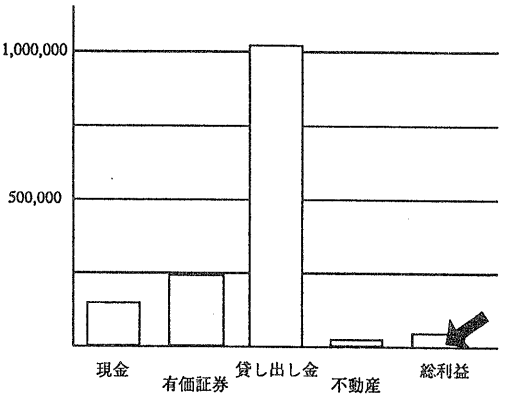

図 2: 予算計画のグラフ表現

図２は貸し出し金の利益率を５．３％にした時グラフ表現である．ここで，利益を以前の状態まで戻したいので，専門家はマウスポインタで総利益を指し，以前の利益になるまでドラッグする．こうすることで，指定された総利益になるように予算は自動的に配分される．ここで強調したいことは，システムに入力した制約を自動的に満たすように配分にしてくれることである．一応，この計画案で目的は達成されている．

つぎに，より良い計画案を模索することにしよう．ここで総利益は満たされたので，マウスポインタで総利益を指し，ダブルクリックする．この操作により，総利益は固定される．同時に対応するグラフの表示属性が変化する（図３）．

また，有価証券に関して一応の目標は達成できたのだが，今以上の額になれば，より良いというときには，有価証券をダブルクリックを２回行なうことで，有価証券の範囲（より大きい）が設定される．

ここで予算項目をマウスで変更する事で，新しい計画案を作成してみる．ここで生成される計画案は総利益が同じで，有価証券の予算額は以前より減ることはない．この二つの計画案から，専門家は失念していた評価に気づき，より良い計画案を得られるかもしれない．

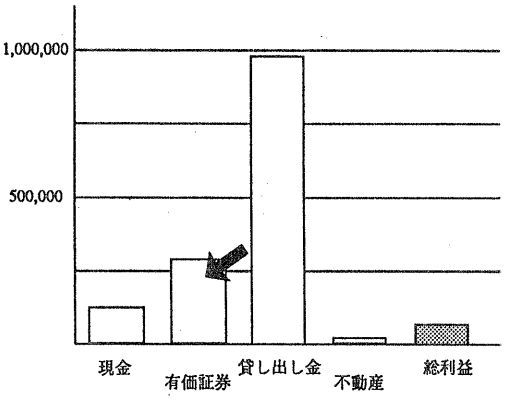

図 3: グラフにおける予算の変更

6 システムの説明

予算計画システムとして，図 4 を提案する．このシステムは計画エンジン部を中心にして考えられている．ルール実行部およびユーザは計画エンジンを動作させることにより，協調的に計画案を作成していく構成になっている．

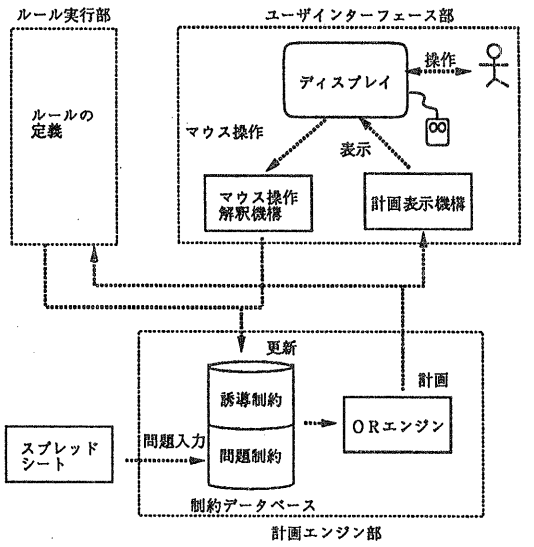

図 4: アーキテクチャ

6.1 計画エンジン

ここでは，予算計画における計画案と制約の関係について述べ，次に制約を満たしながら，計画案を作成するための機構について説明する．

6.1.1 問題制約と計画空間

我々は，節 2 で予算計画における制約について定義した．この制約は，作成される計画案を規定するものであり，これからこの制約を問題制約と呼ぶ．

1 次の等式又は不等式からなる問題制約は，図 5 に示すようにN 次元のユークリッド空間の中の閉じた凸領域になる [1] .

問題制約を満たす，一つの計画案は図 5 の黒点によって表される．計画作成において問題制約を満たすという事は，この計画空間内部の点を保証することであり，我々のシステムにおける計画のステップは，この問題制約から規定される計画空間を移動していることに他ならない．本システムは，グラフ化された計画を専門家が操作することによって，専門家の最終目標の地点へと移動することを目標にしたものである．

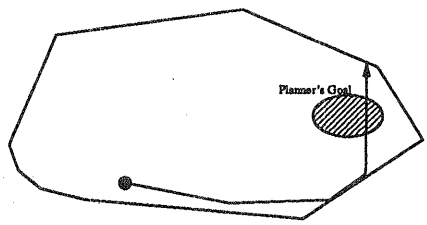

図 5: 計画空間

6.1.2 誘導制約

我々は，誘導制約と言う概念を導入した．誘導制約とは専門家の操作によって生成される，計画案を誘導するための制約である．図 6 を例にして説明する．これは二つの予算項目からなる予算計画の計画空間であり，黒点は一つの計画である．専門家が予算項目 X の値を a から b に変更したとき，X=b という誘導制約が，生成されたと考える．問題制約と誘導制約の交差する領域（太線）は専門家の望む計画案の集合である．

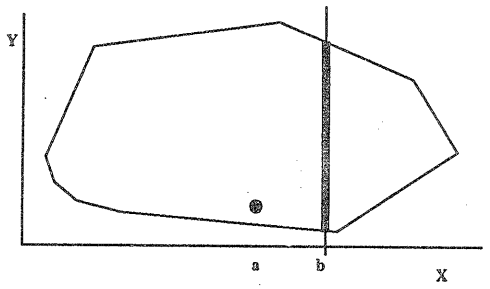

図 6: 誘導制約の例

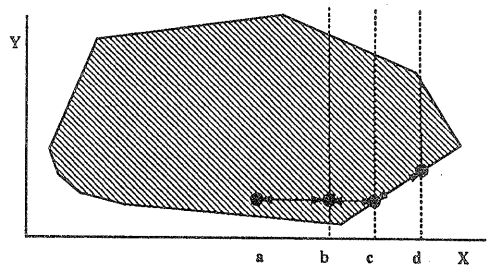

図 7: 計画が戻れる説明

新しい計画案は，この太線中のどの計画案でも構わないが，専門家の感覚に合うのは誘導制約と問題制約の交差する領域の中で，最も現在の計画案に近いものであると考えた．というのは変数X に関して調整したのだから，変数Y は変わらないのが自然だと考えた．

図 6 の場合には，現在の計画案から X=b へと垂線を落した点が新しい計画案となる．

6.1.3 計画エンジンの実装

節 6.1.1 と節 6.1.2 で新しい計画案を求めることの意味について述べた．ここでは計画エンジンの動作を説明しながら，節 3 のシステム要件がどのように実現されているのかについて述べる．

計画案は制約を満たすこと（要件 1 ）は節 6.1.2 の手続きから保証される．つまり新しい計画案は問題制約と誘導制約の交差領域から求められるため，必然的に問題制約を満たす．

節 6.1.2 の説明は変数値の調整に関する誘導制約の役割を示しているが，評価値の調整（要件 2 ）における誘導制約の役割も同様である．評価関数は線形であると定義した事から，評価値の変更は，ある点から超平面への移動と考えられ，その点は一つであることが保証される．変数値の調整の手続きと同様に，評価値の超平面と問題制約の交差領域の中で，現在の点から最も距離の短い点が新しい計画案となる．

要件 3 で述べたように，現在の計画案を中心にして，新しい計画案との間を行き来できなければならない．図 7 は x = a の計画案から x = d までの計画案を誘導制約によって導く動作例である．各ステップについて節 6.1.2 の操作を適用してしまうと x = d から戻る時には点 d から x = a ， x = b ， x = c への垂線を落した点にしか戻れないことになる．本システムでは，移動を始めた点（ x = a ）を基準点として計画案を生成することにした．こうすることで x = d から x = a までの任意の計画案へ戻ることが可能となる．

システム化では問題制約と誘導制約を入れるための制約データベースを作り，問題制約は永続的なもの，誘導制約は一次的なものとしてデータベースに蓄えている．

節 6.1.2 の手続きは 1 次の等式（不等式）である問題制約と誘導制約の交差する領域の中で，現在の計画案に対応する点からの距離を最小にする点（新しい計画案）を求めることである．本システムではこの手続きを 1 次の制約式と 2 次の目的関数からなる 2 次計画問題と捉えて，システム化した．具体的には制約データベースの内容全体を制約として，次の式を目的関数として beale の 2 次計画アルゴリズム [3] で解いている．

$$\min \sum (X_i - a_i)^2 \qquad (1)$$

ここで X_i は変数を表し，a_i は基準点における変数値を表す．この基準点は新たな誘導制約の設定によって，その時点の計画案が基準点として採用される．以上のシステム化により，対話性を実現する処理速度を得ることがプロトタイプにて確認された．

6.2 ユーザインターフェース部

専門家の操作は，ディスプレイ上に表示されている現在の計画を評価し，修正するという意図から発生する．専門家の意図は，マウスの操作として，システムに入力される．マウス操作は誘導制約として翻訳され，誘導制約と問題制約から，計画エンジン

によって新たな計画案が求められ，ディスプレイにグラフとして表示される．以上の操作を対話的に繰り返すことで，計画者はあたかも自分自身が問題を解いているかのように思えれば理想的である [10]．

本システムでは計画案の傾向を制御する（要件5）ために，以下の二つを備えている．

- 変数値の数値の指定．
- 変数の取り得る範囲の指定．

それぞれの操作は，変数に対応するグラフをドラッグすることおよびダブルクリックすることである．変数範囲の設定は，ダブルクリックを繰り返すことで，変数の固定，現在値より大きい，現在値より小さいおよび変数制限の解除の順で設定される．この設定に従ってグラフの表示属性が変化するので，専門家はその設定を確認できる．

ここで発生した操作は，誘導制約に変換される．結局，専門家による計画案を作成するという意図は，まず専門家の操作に置き換えられ，次に操作から誘導制約へと，マウス操作解釈機構によって変換される．図8はその対応を表にしたものである．

また，ユーザインターフェースに求められている要件2および4の実現は節5で述べられた計画手順の概略から明らかであろう．

6.3 ルール実行部

ルール実行部の役割はルーチンワークを専門家の代わりに実行する事である．

専門家の操作から計画エンジンが実行された後，実行の制御がプロダクションシステムに渡り，条件部を満たしたルールが存在するとそのルールを実行する．ルールの処理は計画エンジンを駆動するための処理を記述し，その実行によって計画エンジンを駆動する．この実行が，条件を満たしているルールがなくなるまで繰り返される．

ルールで表現された知識の例を挙げる．専門家はリスクが低い時には，とりあえずリスクが設定した値になるまで，利益を増やすという処理をしていたとしよう．この場合，

```
if   6 ≦ PrimeRate < 7 and
     Risk(PrimeRate) < RISK_AVE
then put ''変更する変数の選択'',
     increase Profit
       until Risk(PrimeRate) ≧ RISK_AVE
```

のように定義できる．ただし，PrimeRate と Profit はそれぞれ公定歩合，利益を表す変数であり，Risk() は計画案の基で公定歩合からリスクを計算する関数，RISK_AVE は経験から得られたリスク設定用の定数，put および increase~until は計画エンジンを操作するためのコマンドで，それぞれ誘導制約の設定および計画の変更を意味する．

7 考察

この節では，制約の定義可能性および節 3.1 で保留した，最終目標への到達可能性についてそれぞれ考察する．

7.1 制約の定義可能性

節 2 に定義した制約の条件について検討する．制約式に関してすべての制約をあらかじめ定義することが可能であると述べた．その理由として，曖昧な制約とは目標達成の概念に関係しているのではないかと考えたからである．目標達成の概念は多目標計画 [2] の概念として紹介されたものである．目標達成とは，達成不可能かもしれない目標が設定され，それを制約として取り込んだものである．このように曖昧な制約は評価に対する目標値から生成される．本論文のシステムでは目標達成の概念をすなおに扱うことが可能である．つまり評価値を操作することで，目標を達成したか否かの判断がつき，達成したのならばその評価値を固定することができるからである．この意味で，本システムは曖昧な制約を扱っているということができる．

7.2 最終目標への到達可能性

我々は，節 2 で評価関数について定義した．そこで評価関数やその順序付けをあらかじめ定義することは難しいと述べたが，専門家の知識としては，すべての評価関数とその順序付けが存在する．ここで次の仮定を置こう．

仮定 複数の評価関数が与えられた時，専門家は総合的な評価関数をそれぞれの評価関数の線形結合によって得ている．

節2とここで述べた仮定は，目標計画法などで仮定していることから比べてみて，妥当であると考えている．

予算計画における評価関数は線形であるから，上記仮定から得られる総合評価関数は線形となる．線形の性質から予算計画では計画空間中に局所最適解は存在しない．節3.1で述べた探索方法は山登り法であるから，局所最適解を持たない問題では必ず最終目標へと到達することができる．

8 おわりに

本論文では予算計画を検討した結果，形式化困難な評価を取り扱うことのできる協調型システムが有効なのではないかと考え，その実現方法について報告した．構築したプロトタイプの特徴は，

1. 計画エンジンが常に線形の制約式を満たした計画案を生成する．
2. 専門家は自身の評価基準に基づいて次の計画の修正を指示できるが，それはシステムによって誘導制約として扱われ，その誘導制約を計画エンジンに渡すことによって修正指示を反映した計画案を得られる．

である．

構築したプロトタイプは，節6で述べたようなルール・ベースは未実装であって，形式化の困難な評価の取り扱いは全て専門家に頼っている．今後の課題は，専門家の知識を獲得していき，エキスパートシステムを構築することである．そのようにして構築したエキスパートシステムは計画の正当性が上記特徴1によって保証されている点が重要であると考えている．

参考文献

[1] S.I.Gass, "Linear Programming", McGraw-Hill,1958,4th Edition, 1975.

[2] M. Zeleny, "Linear Multiobjective Programming", Wpringer-Verlag, 1974.

[3] Beale,E.M.L., Notes on A Comparison of Two Methods of Quadratic Programming, Operations Research, Vol14, 1966.

[4] Barbara,H.R. and Frederich,H.R., A Cognitive Model of Planning, Cognitive Science, Ablex Publishing Corporation, 1979.

[5] 森下ら, 協調型スケジューリングによる製鋼工程スケージューリング・エキスパートシステム, 人工知能学会誌,Vol. 5,No.2,1990.

[6] 金井ら, プランニング業務のためのエキスパートシステム構築環境, 人工知能学会誌,Vol. 5,No.2,1990.

[7] Jaffer, J. and Michaylov, S., Methodology and Implementation of a CLP System, Logic Programming, Vol. 1, pp.196-218, The MIT Press.

[8] Alan Borning, The Programming Language Aspects of ThingLab, a Constraint-Oriented Simulation Laboratory, ACM Transactions on Programming Languages and Systems,Vol 3,No.4,1981.

[9] Gerhard Fischer, Communication Requirements for Cooperative Problem Solving Systems, Information Systems, Vol 15, No. 1, pp.21-36, 1990.

[10] G. Fischer and A.C. Lemke. Construction kits and design environments:steps toward human problem-domain communication, Human-Computer Interaction, Vol 3. pp.179-222, 1988.

[11] David D.Woods, Cognitive Technologies: The Design of Joint Human-Machine Cognitive Systems, AI Magazine, Vol 6,pp.86-92, 1986

意図の分類	意図	操作	誘導制約
変数値の変更	動かす	ドラッグ	等式制約の更新
変数値の範囲の変更	固定する 現在値より大きく（小さく） 制限を除く	ダブルクリック	等式制約の設定 対応する不等号制約の設定 誘導制約の削除

図 8: 操作と誘導制約の対応