ORの実践　－理論と産学連携－

㈱ 数理モデリング研究所
野末 尚次
URL: www.math-model.co.jp

Math-Model Research Inc.

ORの実践は、なぜ上手く行かないのか？

1. 問題の把握
2. 適用可能なORモデルの探索
3. モデル化
4. 解の探索法の適用
5. 解の評価
6. 問題への解の適用
7. 状況の変化に対応
8. 問題解決？

問題は徐々に分かる

モデルを見つけられない

モデルが単純すぎる

データが無い
答えが出ない

解の質が悪い／評価が困難

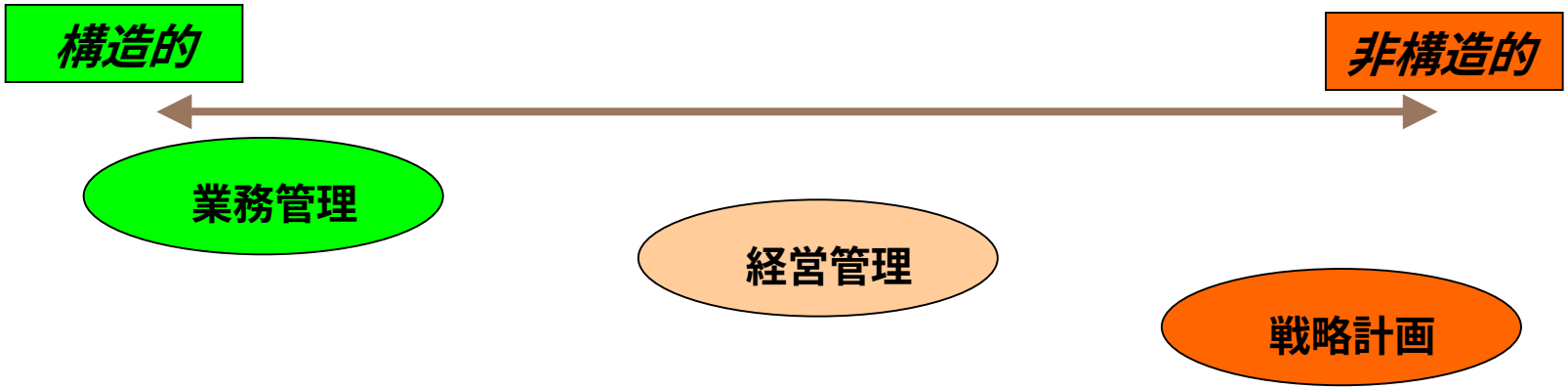
実用上の制約を満たさない

解の連続性が無い
モデルが改修できない

ORは役に立たない？

ORのモデル･理論は、実践的か？

意思決定問題の分類



[構造的]

・問題の本質と構造が明確

・解法も確立

・定型的で繰り返し

[非構造的]

・問題の構造が複雑で不明確

・解法が確立していない

・非定型的

意思決定とDecision Support System (DSS)

[Simon]　人間の意思決定過程は、次の3段階で構成される。

(1) **発見過程**:

- 関連情報の収集
- それに基づいた問題点の発見

(2) **設計過程**:

- 問題の定式化
- 可能解の生成
- 実現可能性のチェックによる代替案の設定

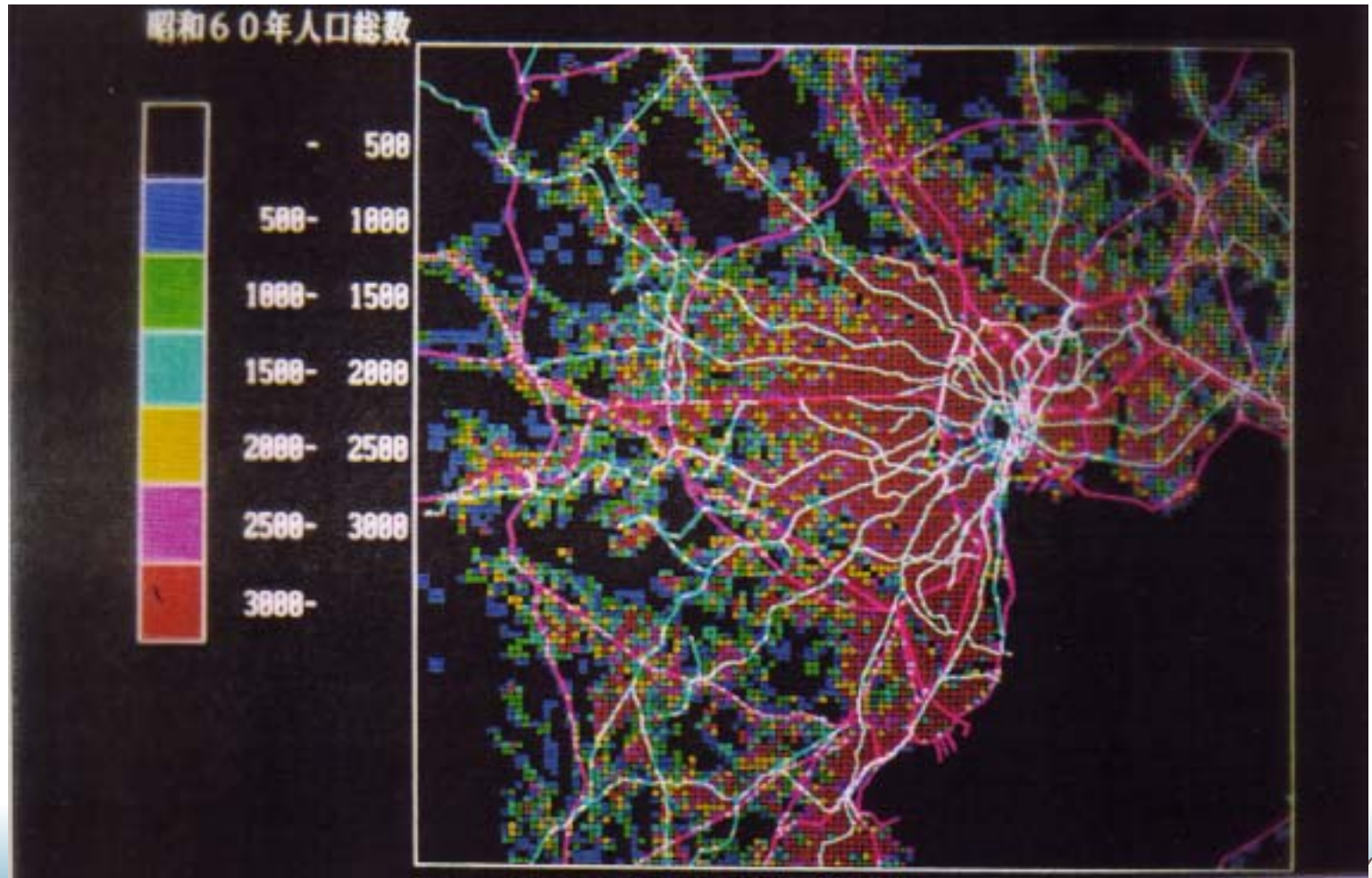
(3) **選択過程**:

- 複数の代替案の中から1つの案の選択
- 実行案の作成

DSSは、これらの3段階を通じて、人間との共同作業を行うシステムである

人間の知識＋直感力＋総合力 (1)

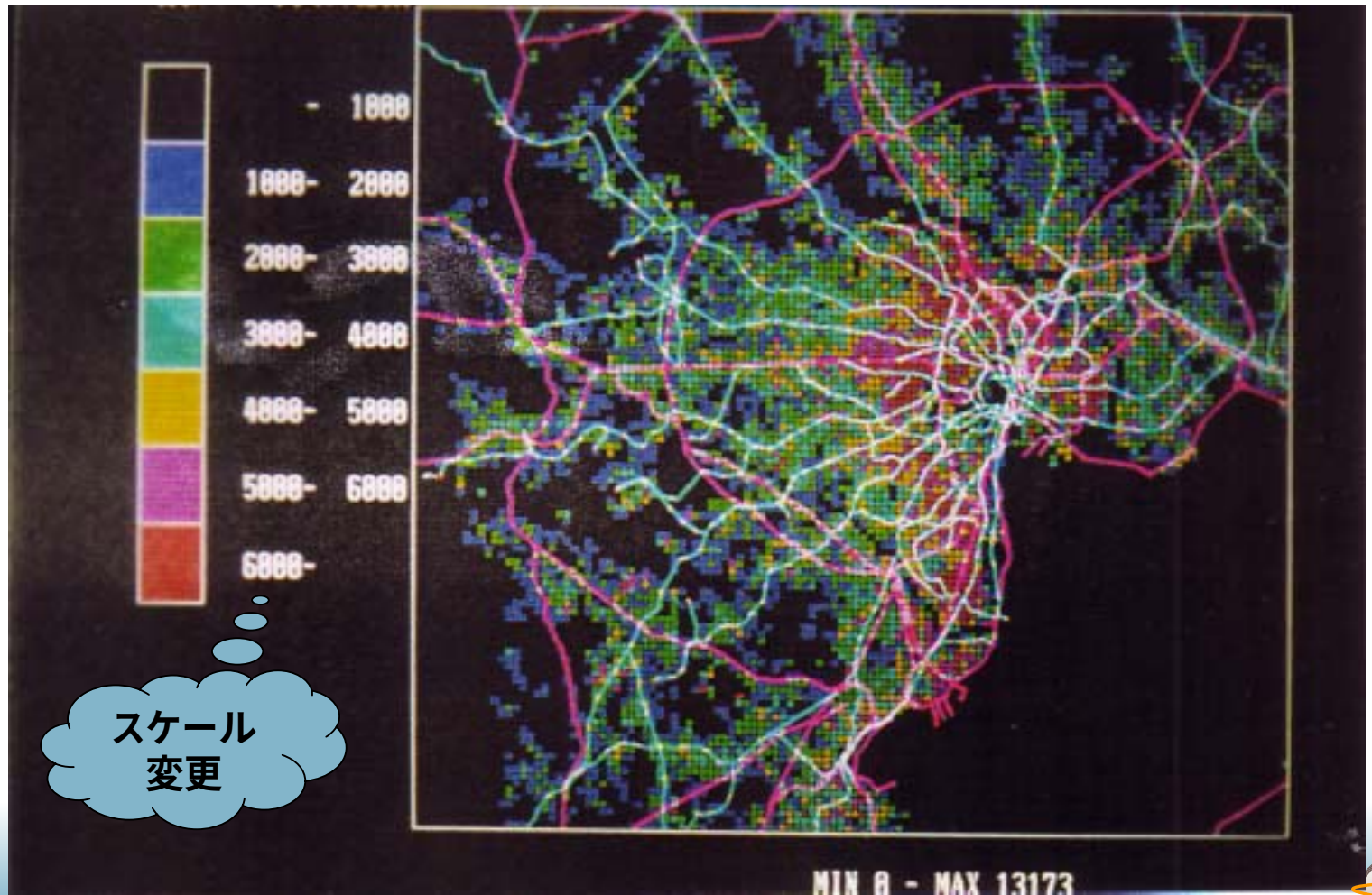
首都圏の昭和６０年の人口分布：地理情報処理システムTRAMPS＋国勢調査メッシュデー



Math-Model Research Inc.

人間の知識＋直感力＋総合力 (2)

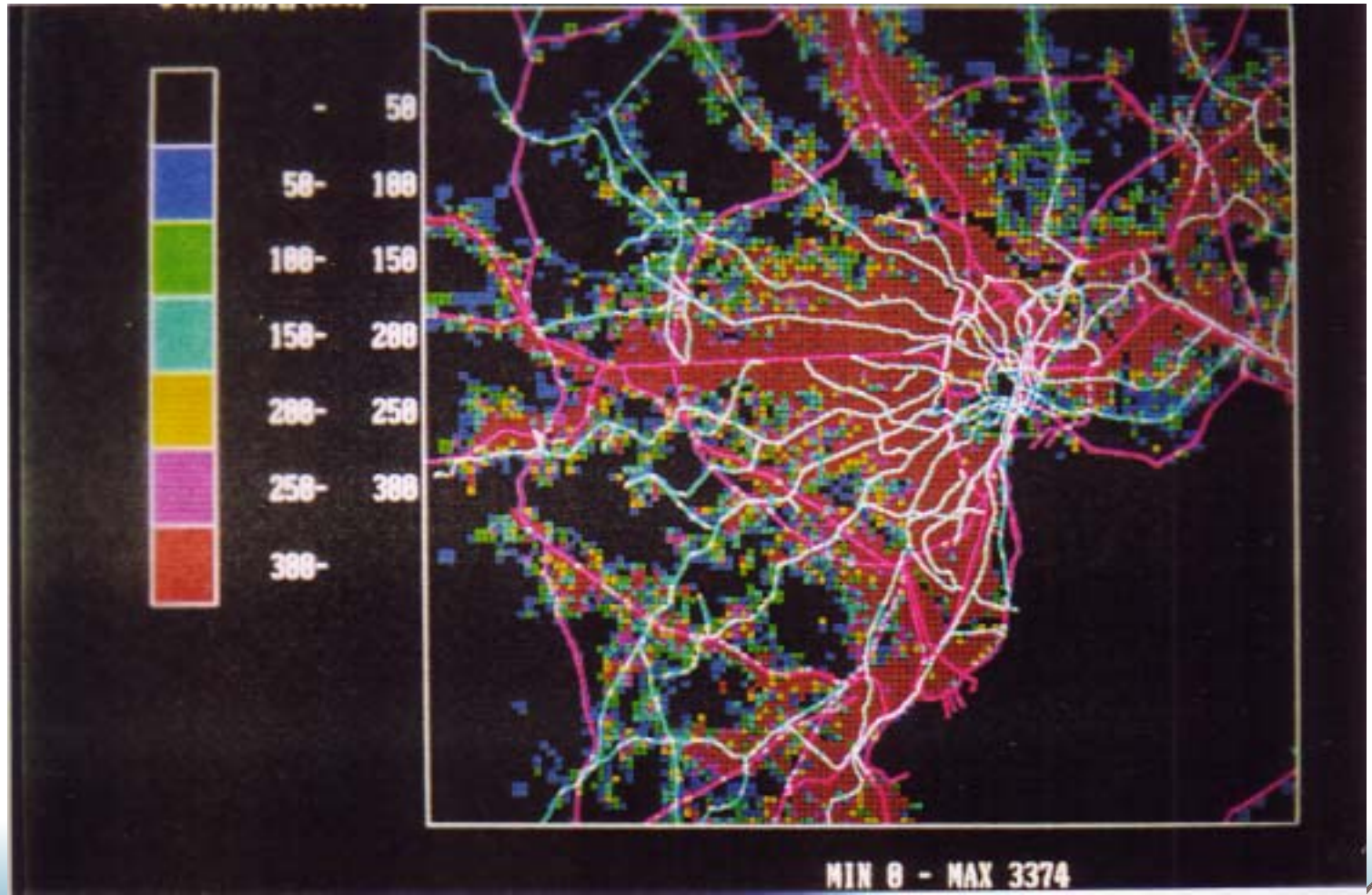
首都圏の昭和６０年の人口分布：地理情報処理システムTRAMPS＋国勢調査メッシュデー



Math-Model Research Inc.

人間の知識＋直感力＋総合力 (3)

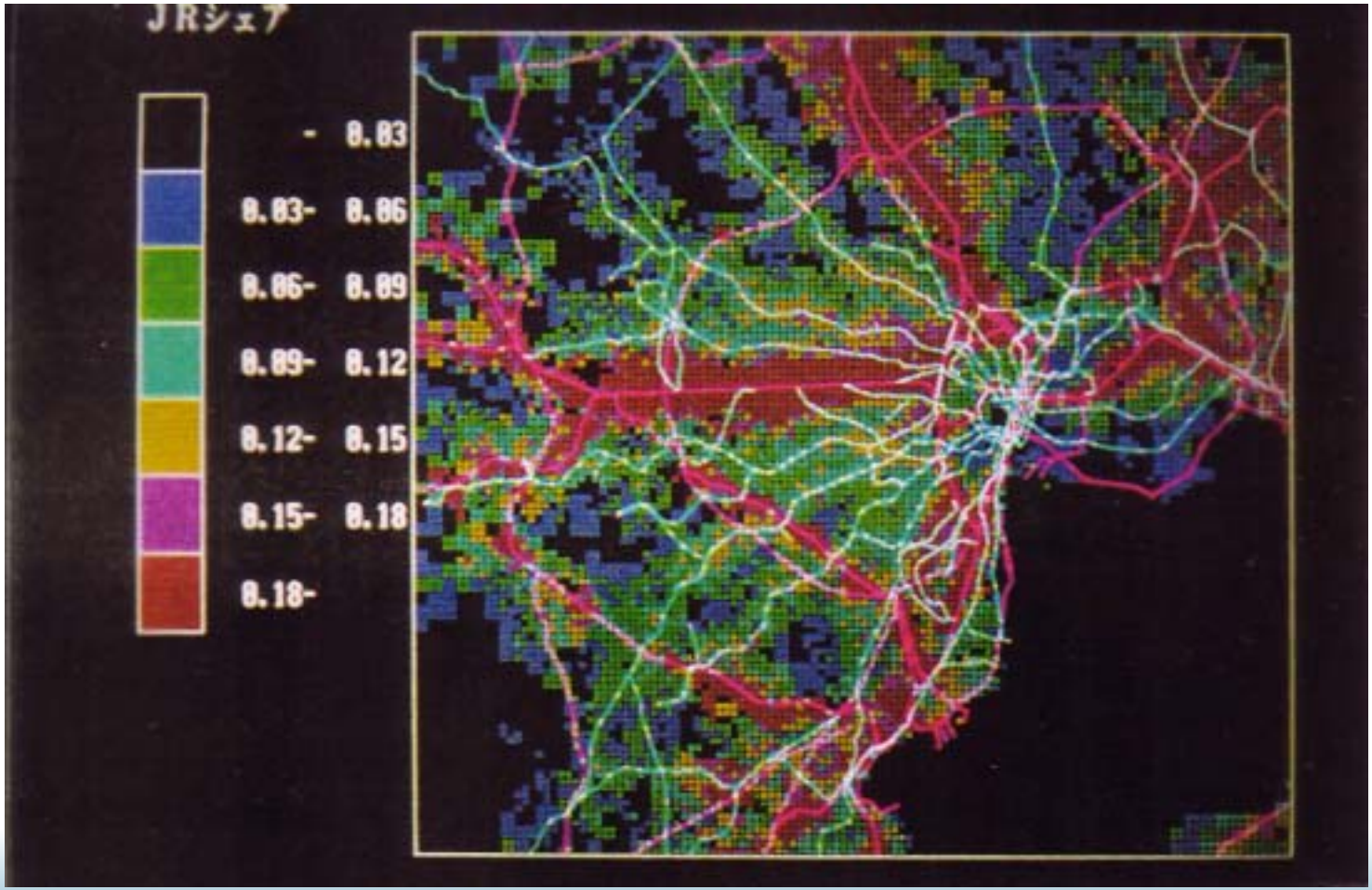
国鉄を利用する通勤者の居住地分布 (昭和６０年) :



Math-Model Research Inc.

人間の知識＋直感力＋総合力 (4)

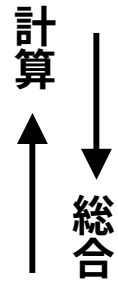
国鉄を利用する通勤者のシェア分布 (昭和60年) :



Math-Model Research Inc.

意思決定のプロセス：Generalized Problem Processing (GPSS)

	問題発見	代替案作成	代替案選択
データ収集	1	4	7
データ加工	2	5	8
評価･選択	3	6	9



(1) データ収集
(2) 問題認識
(3) 概念モデルの構成
(4) 妥当性の検証
(5) 分析
(6) 解決策の導出
(7) 実行可能性の検証
(8) 実施案の作成
(9) 実施案の提案

実用規模の計画問題の特徴

計画問題

機械・人間・材料・在庫・納期等の制約条件の下で、

- 制約条件を満たす計画案（実行可能案）を作成する。
- 実行可能案の中で、最も効率の良い案を求める。

- どのようなタイプの制約条件があるか？
  - 絶対守らなければならない制約（ハードな制約）
  - ペナルティは有るが、緩和が可能な制約（ソフトな制約）
- 評価指標はなにか？
  - 実行可能解が得られれば良い（制約充足問題）。
  - 実行可能解の中から、（準）最適解を求める（最適化問題）。

計画問題へのアプローチ　(開発手法)

- 計画問題に対する2つのアプローチ
  - 手続的なアプローチ (従来のプログラム)
  - 宣言的なアプローチ (制約の定義と解探索の分離)

百足もいるよ！

つる･かめ算の計算

鶴と亀が足して5匹、足の合計が14本の時、鶴と亀の数は？

手続的アプローチ

1) 全部鶴と仮定すると、足は10本

2) 余った足の数　14－10＝4本

3) 亀の数　＝　4本／2　＝2匹

4) 鶴の数＝5－2＝3匹

宣言的アプローチ

1) 鶴と亀の数を　X、Yとする

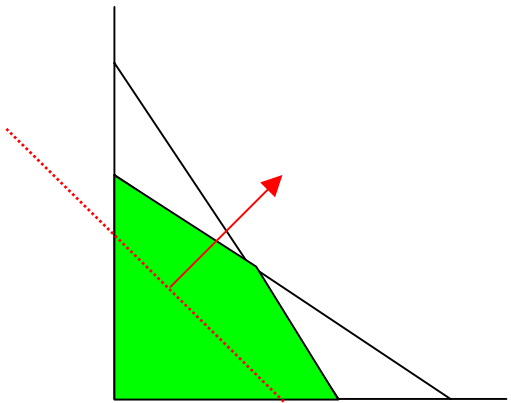
2) X＋Y＝5　2X＋4Y＝14　(定義)

3) 連立方程式を解く　(探索)

4)　X＝3　Y＝2

最適化問題の分類 (質的相違)

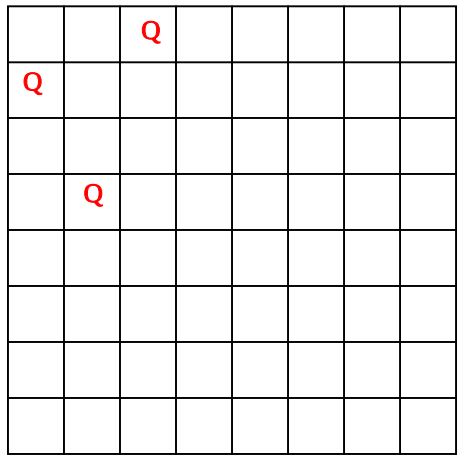
資源配分問題



$$2x + 3y \leq 12$$
$$3x + 2y \leq 12$$
$$x \geq 0,\ y \geq 0$$

$$x + y \rightarrow Max$$

8-Queen配置問題



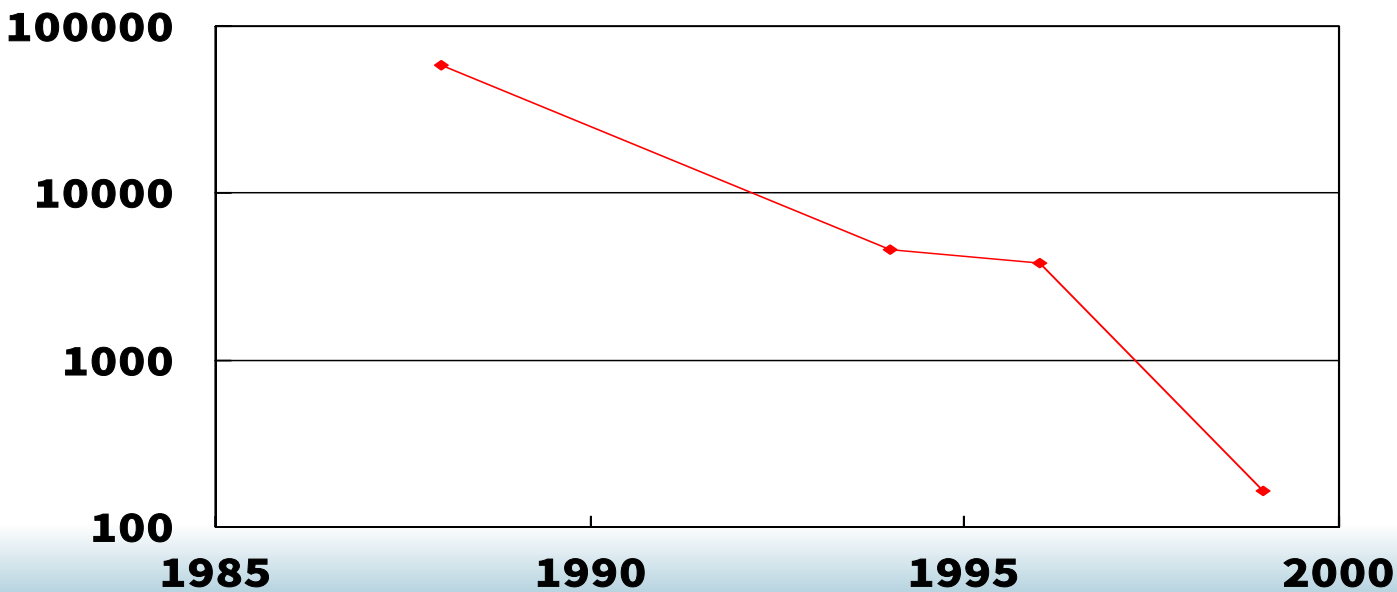
$$x[i] \neq x[j],\ for \quad i \neq j$$
$$x[i]+i \neq x[j]+j,\ for \quad i \neq j$$
$$x[i]-i \neq x[j]-j,\ for \quad i \neq j$$

計画用ソフトウェアの進歩

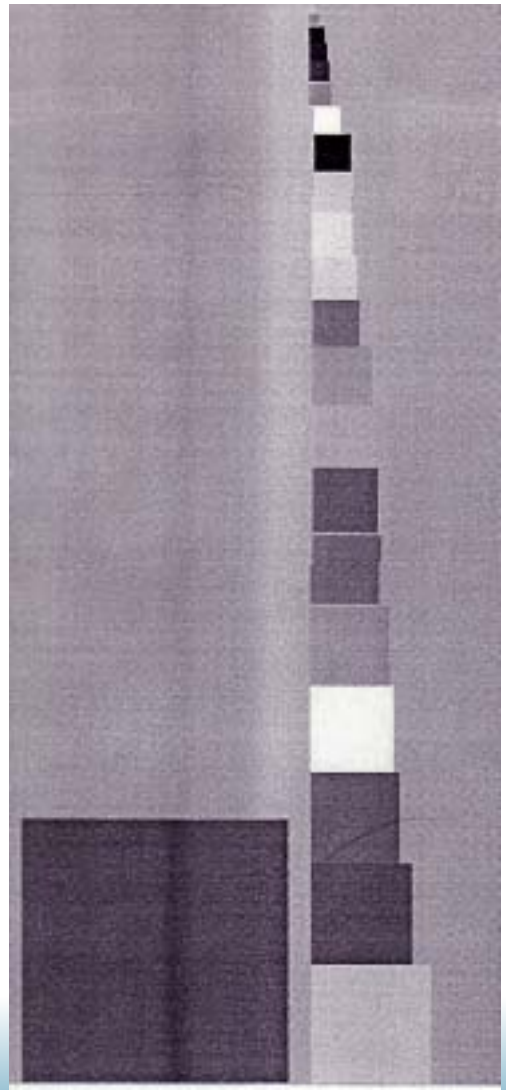
- ＬＰのヴァージョン別実行速度の比較（ILOG社CPLEX）

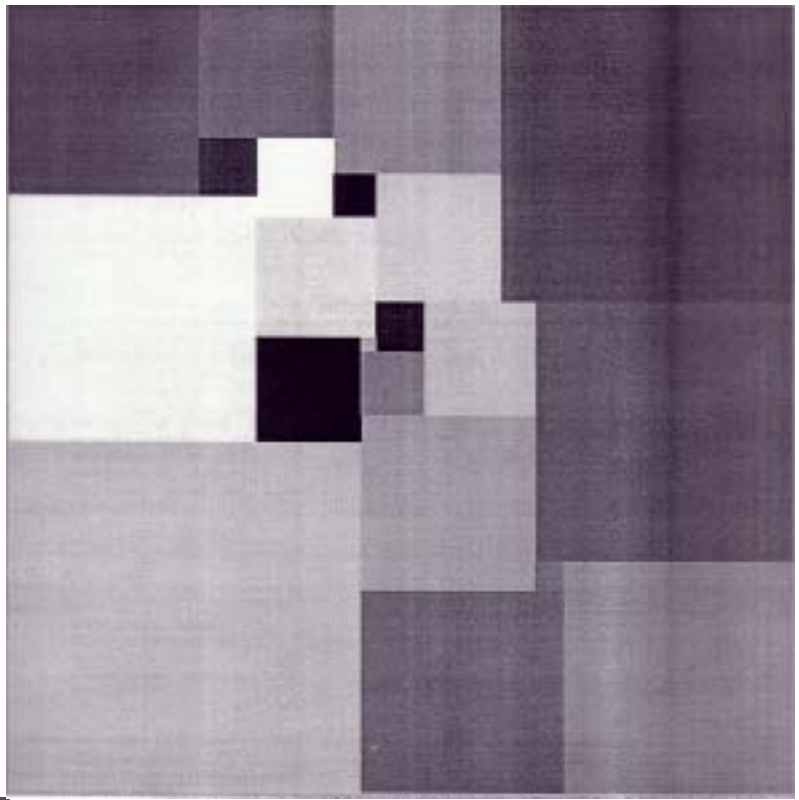
  （CPLEX1.0～CPLEX6．0）

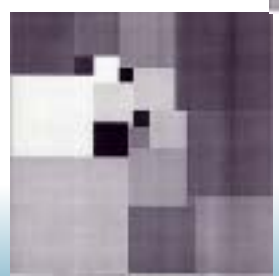
  - 問題規模：　４９９４４行，１７７６２８列　計算時間（秒）
  - 計算時間：　単位（秒），マシーンは同一



宣言的アプローチのソフトウェア（制約プログラミング）パッケージの出現
ILOG社のSolver／Scheduler、COSYTEC社のCHIP







計画問題へのアプローチ　(整合性の維持)

エキスパート･システムは、なぜ失敗したか

- 知識工学の新しい手法
  - 局所的なルールにより熟練者の知識を組織化する。

- 診断系のシステム開発で大きな成果

- 計画系のシステム開発に適用
  - 多くの制約がある実用規模の開発は、実質的に失敗

- 局所的な情報で有効と思われる方策も、
  トータルに見ると有害な場合がある。
  - 制約条件全体を考慮した方策が選択されていない

宣言的アプローチ：制約論理、制約プログラミング

- 論理言語 (Logic Programming) ：Prolog
  - 論理的な制約条件を宣言的に記述可能
  - ユニフィケーションとバックトラックによる探索

  ⇒ 計算時間がかかる。
- 制約充足問題 (Constraint Satisfaction)
  - 有限領域を対象とした効率的な探索方式／整合性と領域縮小

  ⇒ 問題の定式化に自由度ない。
- 平行処理 (Parallel Processing)
  - 複数のプロセスにより計算状況を監視／動的な制約伝播

↓

制約論理言語：CHIP (Constraint Handling in Prolog)

↓

制約ツールキット：ILOG (C++で制約ベースの処理を実現)

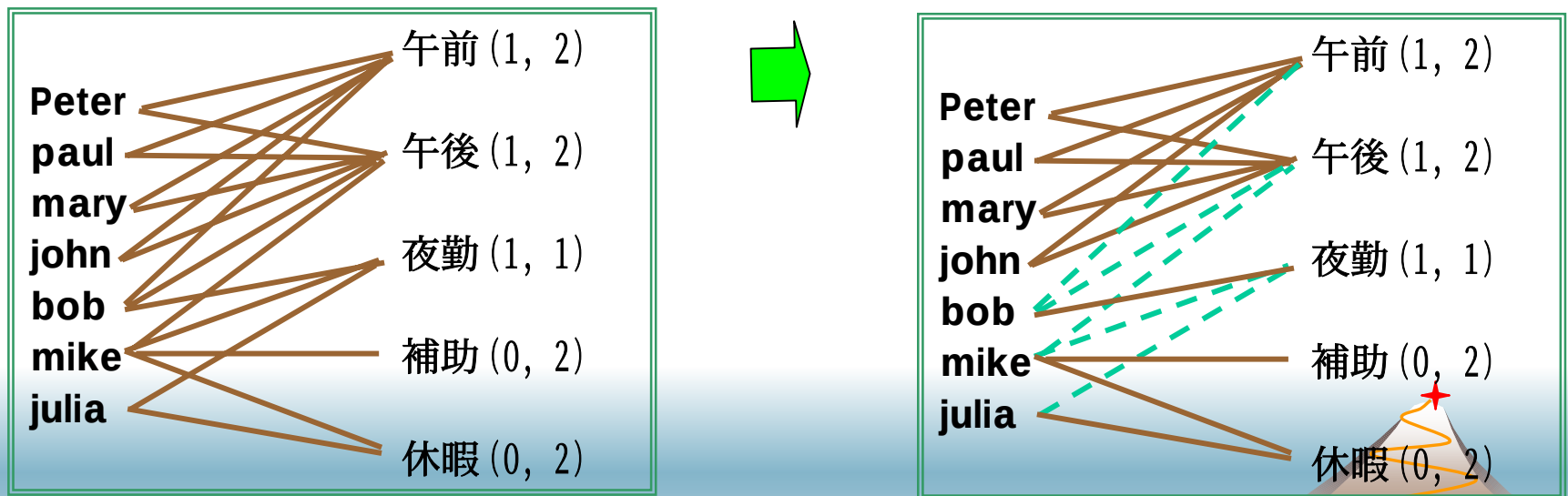
制約プログラムの例 (部分)

```
IlcManager m(IlcNoEdit);                                    マネージャの定義

IlcIntVarArray x(m, nqueen, 0, nqueen-1),                   変数の定義
                    x1(m, nqueen),    x2(m, nqueen);
IlcInt i;
for (i = 0; i < nqueen; i++) {
          x1[i] = x[i] + i;    x2[i] = x[i] - i;  }         変数間の制約

m.add( IlcAllDiff(x) );                                     制約条件の定義
m.add( IlcAllDiff(x1) );
m.add( IlcAllDiff(x2) );

m.add( IlcGenerate(x, IlcChooseMinSizeMin) );               探索法の指定

if ( m.nextSolution() ) {                                   解の探索
          for (i=0; i < nqueen ; i++ )    m.out() << x[i].getValue()  << " ";
          m.out() << endl;
}  else m.out() << "No solution" << endl;

m.printInformation();
m.end();                                                    Version 4
```

緩和問題による大域的な制約伝播

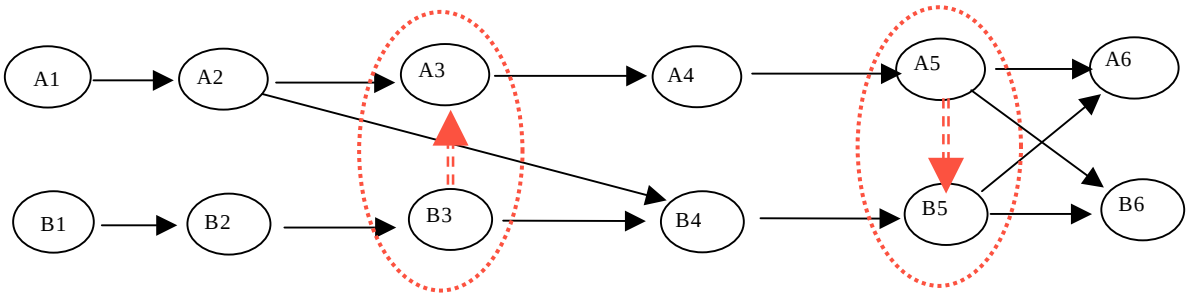
- 制約伝播の不完全性
  - 制約（論理）プログラミングの固有の制約伝播だけでは、複雑な制約条件に対して、完全な制約伝播を行うことは、計算時間上、不可能である。
- 大域的な制約条件の導入
  - オペレーションズ・リサーチ等の理論を応用することにより、効率良く制約伝播が可能な汎用の大域的制約条件の導入　－＞　メーカーの差異化
- 例：勤務計画で、勤務毎の人数の制約：　勤務（最小、最大）

最大フロー問題を利用した制約伝播：　J.　Regin　(ILOG)

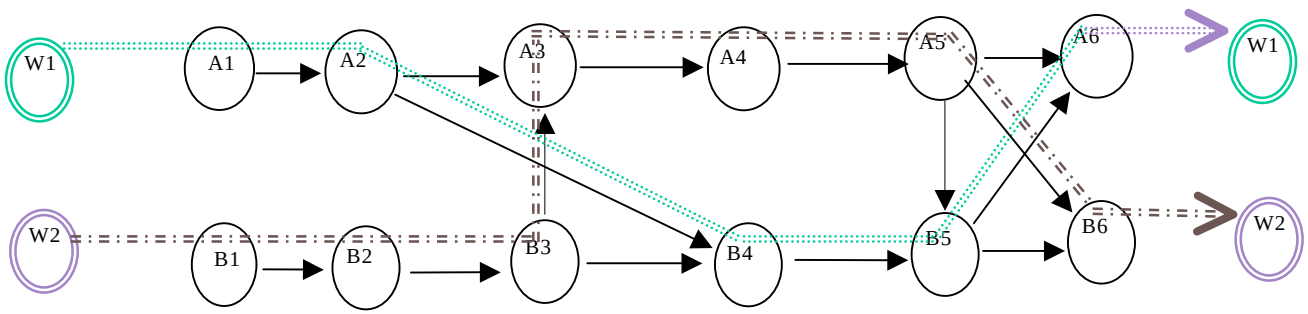


代表的な大域的制約条件とモデル

1) 各作業者の仕事の順序が与えられているが、仕事間の優先順序やリソースの競合があるモデル



2) 順序付けられた仕事に対して、作業者を割り当てるモデル



3) 日々に必要な資格別の要員数、及び、各作業員の勤務種別の時系列に対する制約のモデル

氏名	月	火	水	木	金	土	日
A	N	R	N	D	N	R	R
B	D	N					
C	R	D					
日勤	1	1	2	1	1	1	0
夜勤	1	1	1	1	1	1	1

例：スケジューリング問題の制約条件定義

- 変数の定義：

  $X[s]$：番号ｓの仕事を行った作業者が次ぎに行う仕事の番号

  $T[s]$：番号ｓの仕事の開始時刻

  $Y[s]$：番号ｓの仕事を行った作業者の番号

- 制約条件の定義：

$X[s] \in \mathrm{Suc}[s]$	$s \in N$	許された後続作業の選択
$Y[s] \in P$	$s \in N$	作業者は、限定
$Y[X[s]] = Y[s]$	$s \in N$	続行作業は、同じ人
$T[X[s]] = T[s] + d[s]$	$s \in N$	次ぎの開始時刻の計算
$X[s] \neq X[u]$	$s \neq u \quad s,u \in N$	直前の仕事は、1つだけ
$a_s^{X[s]} \leqq T[s] \leqq b_s^{X[s]}$		開始時刻の制約
$X[15]=1, X[16]=2, X[17]=3, X[18]=4$		開始と終了の一致

非常にコンパクトな表現が可能

制約ベースのモデル化

(1) **発見過程**:

- 制約オブジェクト・モデルの構築　(ユーザと独立した論理的作業)
- 制約条件の抽出　(ユーザの嗜好によらない制約条件の抽出が可能)
- 評価基準の抽出　(多様な価値基準に適応化)

(2) **設計過程**:

- 制約モデルの定式化　(制約の追加・削除が容易)
- 可能解の生成　(既存の計画に近い代替案の生成が可能)
- 緩和問題を利用した非効率な代替案の削除　(産学連携が可能)

(3) **選択過程**:

- 複数の代替案の評価値抽出と最適案の選択　(ユーザの価値基準に適応)
- 実行案の作成　(GUIを利用した確認)

私のOR実践例

・交通計画関係
- 地理情報処理(GIS)と国勢調査データ
- GIS＋交通需要予測モデル
  ＋代替経路生成＋幹線純流動調査
  ＋仮想質問+主観的データ
- GIS＋リモートセンシング
- 時系列データ＋需要予測＋GUI
- ネットワーク(k－thパス)
- 知識工学＋数理計画

・輸送計画関係
- 制約ベースの計画ロジック
- 制約論理と制約プログラミング

- 運賃規則の宣言的表現＋Solverの開発

・信頼性管理関係
- 時系列データ＋磨耗予測

・商圏分析のDSS
・バス路線計画のDSS
・全国新幹線網計画のDSS
・MAGLEV計画のDSS
・鉄道路線計画のDSS
・列車設定計画のDSS
・鉄道運賃経路生成システム
・列車乗継案内システム

宣言型汎用システム開発

・列車ダイヤ作成システムの開発
・新幹線運転整理計画のDSS
・車両運用計画のDSS
・勤務割当計画のDSS
・乗務員交番表計画のDSS

・宣言型運賃計算システムの開発

・車載消耗品管理システムの揮発

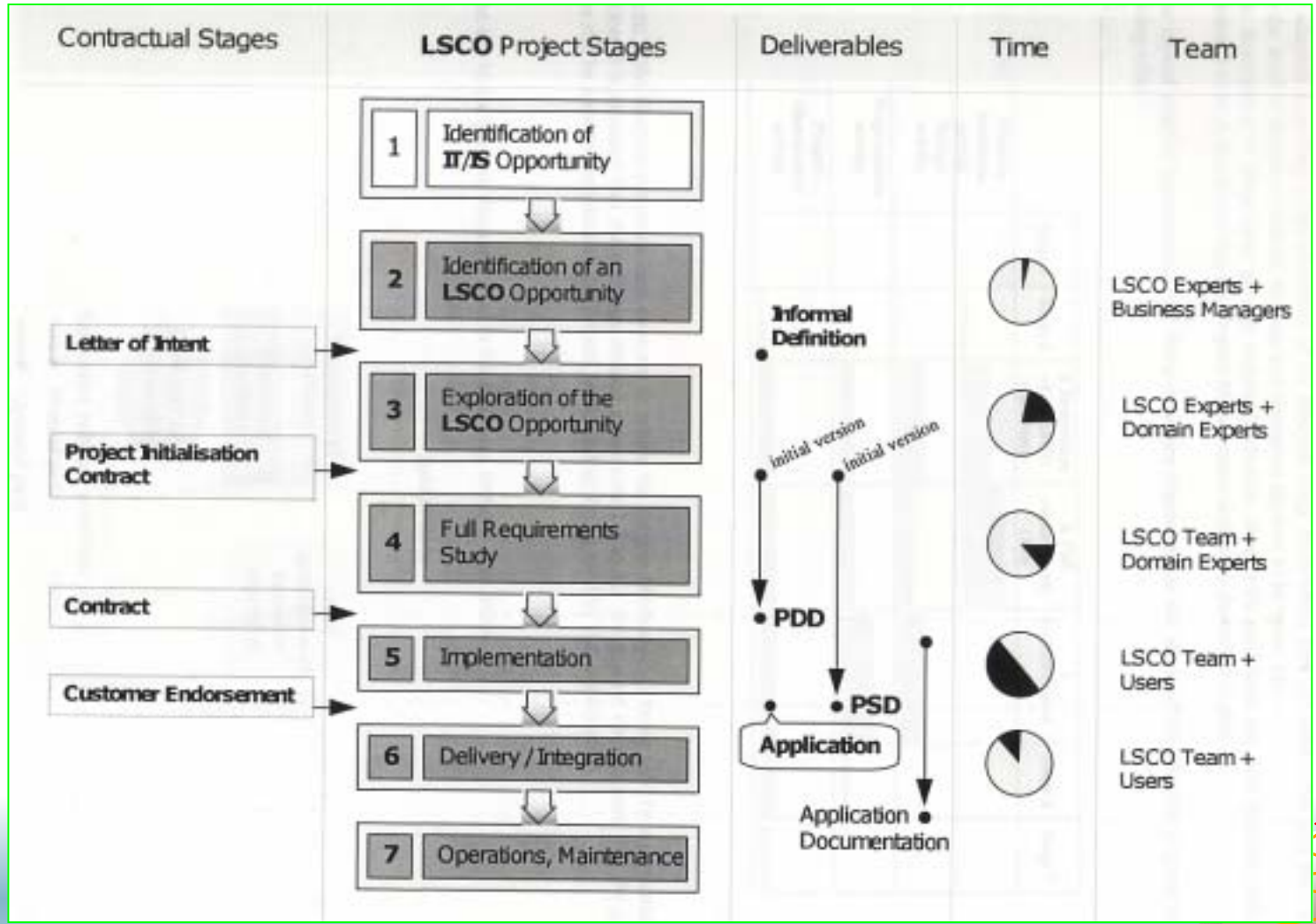
Math-Model Research Inc.

学会への注文

産学連携

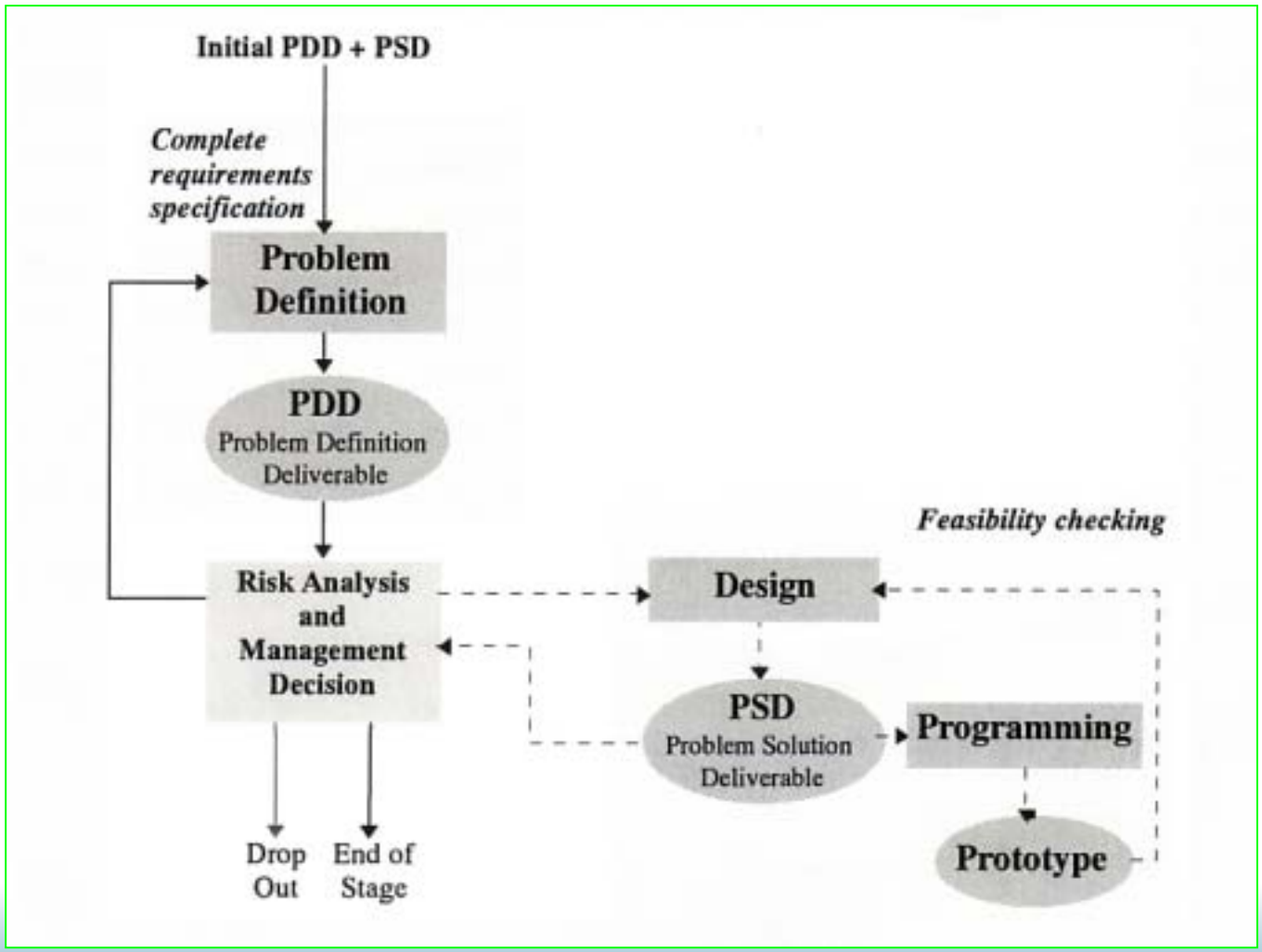
計画系システム開発のプロジェクト管理：CHIC-2

Creating Hybrid Solutions for Industry and Commerce



Math-Model Research Inc.

計画系システム開発のプロジェクト管理：リスク分析



計画系システムのカテゴリー

Table 4.3:Examples of LSCO applications

Field	Application	Category	Classification
Airline industry	Crew scheduling Fleet scheduling Shift planning	Set covering Set partitioning Network flow Assignment	var: Integer constr: Linear problem: Constrained Data: Deterministic
Production manufacturing	Cutting stock	Knapsack Bin packing	var: Mixed integer constr: Non-linear problem: Constrained Data: Deterministic
Production scheduling	Car sequencing Resource allocation Task scheduling	Resource constrained project scheduling	var: Integer constr: Non-linear problem: Constrained Data: Deterministic
Transportation industry	Facility location Vehicle routing Tour problems	Set covering TSP	var: Integer constr: Linear problem: Constrained Data: Deterministic
Telecommunication and network industry	Network flow Network optimisation	Matching	var: Integer constr: Linear problem: Constrained Data: Stochastic
Personnel scheduling	Time-tabling Work force scheduling	Assignment Matching flow	var: Integer constr: Linear problem: Constrained Data: Deterministic
Finance	Portfolio optimisation	Knapsack	var: Continuous constr: Quadratic problem: Constrained Data: Stochastic

問題別アルゴリズムの評価 (1)

Problem category	Do's	Don'ts
Set covering, set partitioning including possibly some additional knapsack constraints	Mixed Integer Programming Column generation, for very combinatorial problems, allows an efficient technical decomposition	
Set selection (coverage)	LP, CP with partial search (LDS) local optimisation	Pure CP
Scheduling Disjunctive scheduling (single tasks)	B&B, constraint propagation using literature shaving tabu heuristics	LP, genetic algorithms, simulated annealing
Scheduling with complex resources and tasks	Read Demeulemeester' thesis (Demeulemeester 92), Chronological backtrack (B & B), resource histograms, partial search	Pure CP-LP
Production scheduling	Medium/long range (resources allocation with constraints of maximum number of changes in a period of time), big size (multi-product, multi-machines), linear processes : MIP can bring satisfactory solutions, combined sometimes with heuristics for post-processing.	Use of "pure" approaches

問題別アルゴリズムの評価（2）

	Short range problem, including fine set up time, non linear routes (job-shop problems), small size : CP + heuristics. If big size, decompose problems	
Time-tabling Fixed set of activities	is the problem hard ? global constraints, flow algorithms. LP model, local optimisation	Pure CP (local propagation for hard problems)
Variable schedules	LP, tabu, coverage techniques, column generation, CP for schedules global analysis	Pure CP Simple heuristics
Staff scheduling	Shifts construction: use MIP with set covering/partitioning approach. Roster grids construction: CP and heuristics. Decompose and use hybrid approaches	Don't try to solve the whole problem in one round with one approach.
Matching Assignment, resource allocation	Use adhoc algorithm (matching flows) or LP (simpler) search for good model relaxation -> lower bounds local optimisation	pure CP
Assignment problems	Big and linear : MIP Small and non linear: CP or hybrid	
Routing problems Travelling Salesman Problem	CP for small complex problems only, local opt by default, B&Cut is an option bibliographical search !	CP, LP without proper expertise
Vehicle routing problems	Local optimisation	CP only
Tour problems	Columns generation MIP approaches proved to optimally solve such problems. Heuristics, although sub-optimal, proved to solve them very efficiently (ex: TSP with hundreds of cities in some seconds)	If max duration constraints, don't use network flow approach with LP
Flow problems Network optimisation	LP models, flows, global analysis P. good heuristics & local opt	pure CP, genetic algorithms, LP only
Flow optimisation problems including Boolean choices (multi-periodic, multi-products with inventories and intermediate processes)	Linear programming, MIP	Heuristics, CP, ...
Melting problems	Pure LP (MIP)	Others
Cutting problems	1 dimension : MIP can be OK. Several dimensions (2 or 3) : simulated annealing proved to work in some cases.	MIP for more than 2 dimension problems

ORの実践のためには？