次世代のAIを実現するLocalSolver 8.0

2018年9月20日

※宮崎　知明（MSI株式会社）

目次

0. はじめに

コンピュータの飛躍的な性能向上により、リアルタイムで意思決定、判断をすることが可能になり、空前のAIブームを迎えようとしている。

AI、IoT、Big Dataのキーワードのもと、必要な情報を必要なタイミングで得ることができ、あらゆる分野で情報活用が可能となりつつある。

AIによる自動化は以下に大別できる。

- **センサと連動した自動運転制御**（自動車の安全装置、プラント自動制御など）
- **大量の実績データを統計処理し、統計確率にもとづいたオペレーションの自動化**（事務処理の代替など）
- **人間の思考パターンの高速シミュレーション**（ゲーム、ディーププランニング等）

最適化システムの歴史

- **従来型の数理計画法システム**
  - 理論的に最適解を求めるアプローチ
  - 大規模問題になると現実的な時間では解けない
  - 非線形問題に弱い
- **ヒューリスティックスプログラムの出現**
  - 人間の行動を計算機で実現
  - AIとして発展（一種の自動化、最適化を実現）
- **メタヒューリスティックス解法の出現**
  - ヒューリスティックスを汎用的な解法に
  - 問題対応型で様々な解法が出現
    （Greedy algorithm、 Local search、GA・・・）
  - 実用的な最適解を試行

次世代AIとは

現状のAIは統計予測をベースとした統計理論による確率論的なものや、センサ等による情報に対する対処方法がきまっていることが基本であり、意思決定理論が確定していることが重要である。

これに対して、**新しい事象でも対応できるようにする（最適解を導く）**ことが次世代AIである。

一言で言えば、意思決定モデルを作成し、最適化手法を組み込むことにより、実績にない事象に対しても、さらなる効率化、自動化を可能することである。

- オペレーション関連の自動化には、AIによる確率論をベースとした自動化（大規模データ、データモデル等）だけでは不十分であり、プロセスモデルと組み合わせた実用に供する大規模組み合わせ最適化を行う最適化システムがますます重要となっている。
  従来の数理計画法システムでは最近の大規模かつ複雑な最適化要求に答えられていないのが現状。

- 従来の方法（MIP：混合整数計画法）では解けない**大規模組合せ 最適化問題（800万以上のパターンから最適な組み合わせを求める）**を実用的に解くことができる

汎用的な最適化ソフトが出現　⇒　LocalSolver

大規模組み合わせ最適化問題で拡がるソリューション分野

従来、大規模問題（以下に例を示す）は、探索空間が離散的であるもしくは離散的なもので表現できる問題であり、混合整数計画問題として定式化はできたが、実用的に解くことは難しかった。

LocalSolverを使えば実用的に解くことができる。

集合，順序，割当て，グラフ，論理，整数など離散的な構造を持つ現実世界の問題は大規模組合せ最適化問題となる。

- **車両の優先順位付け（組立）**問題
- **裁断計画**問題 (フィルムなど)
- **SCM問題** (製造－輸送－在庫－販売など)
- **最短路問題** (カーナビのルート検索など)
- **ネットワーク**問題 (交通網，通信網，電気、ガスなどの設計)
- **配送計画**問題 (宅配便，店舗への商品配送，ゴミ収集など)
- **施設配置**問題 (工場，店舗，公共施設などの配置など)
- 人員**スケジューリング**問題 (乗務員・看護師の勤務表，時間割の作成など)
- 機械**スケジューリング**問題 (工場の運転計画、装置稼働計画など)

MIPLIBでのベンチマーク結果

Instances	Status	Variables	LocalSolver 3.1	Gurobi 5.5	Cplex 12.4	Optimum
opm2-z10-s2	hard	6,250	* -25,719	-19,601	-18,539	-33,826
opm2-z11-s8	hard	8,019	* -33,028	-21,661	-18,883	-43,485
opm2-z12-s14	hard	10,800	* -46,957	-11,994	-36,469	-64,291
opm2-z12-s7	hard	10,800	* -46,034	-12,375	-30,887	-65,514
pb6	hard	462	-62	-62	-62	-63
queens-30	hard	900	-38	-36	-39	-40
dc11	open	37,297	11,100,000	21,300,000	1,840,402	unknown
ds-big	open	6,020	9,844	62,520	5,256	unknown
ex1010-pi	open	25,200	249	251	247	unknown
ivu06-big	open	1,812,044	* 479	9,416	678	unknown
ivu52	open	1,423,438	4,907	16,880	3,285	unknown
mining	open	753,404	* -65,720,600	902,969,000	no solution	unknown
pb-simp-nonunif	open	23,848	* 90	140	94	unknown
ramos3	open	2,187	* 223	274	267	unknown
rmine14	open	32,205	* -3469	-170	-968	unknown
rmine21	open	162,547	* -3657	-184	no solution	unknown
rmine25	open	326,599	* -3052	-161	no solution	unknown
siena1	open	13,741	256,620,000	315,186,152	54,820,419	unknown
sts405	open	405	342	342	354	unknown
sts729	open	709	648	648	665	unknown

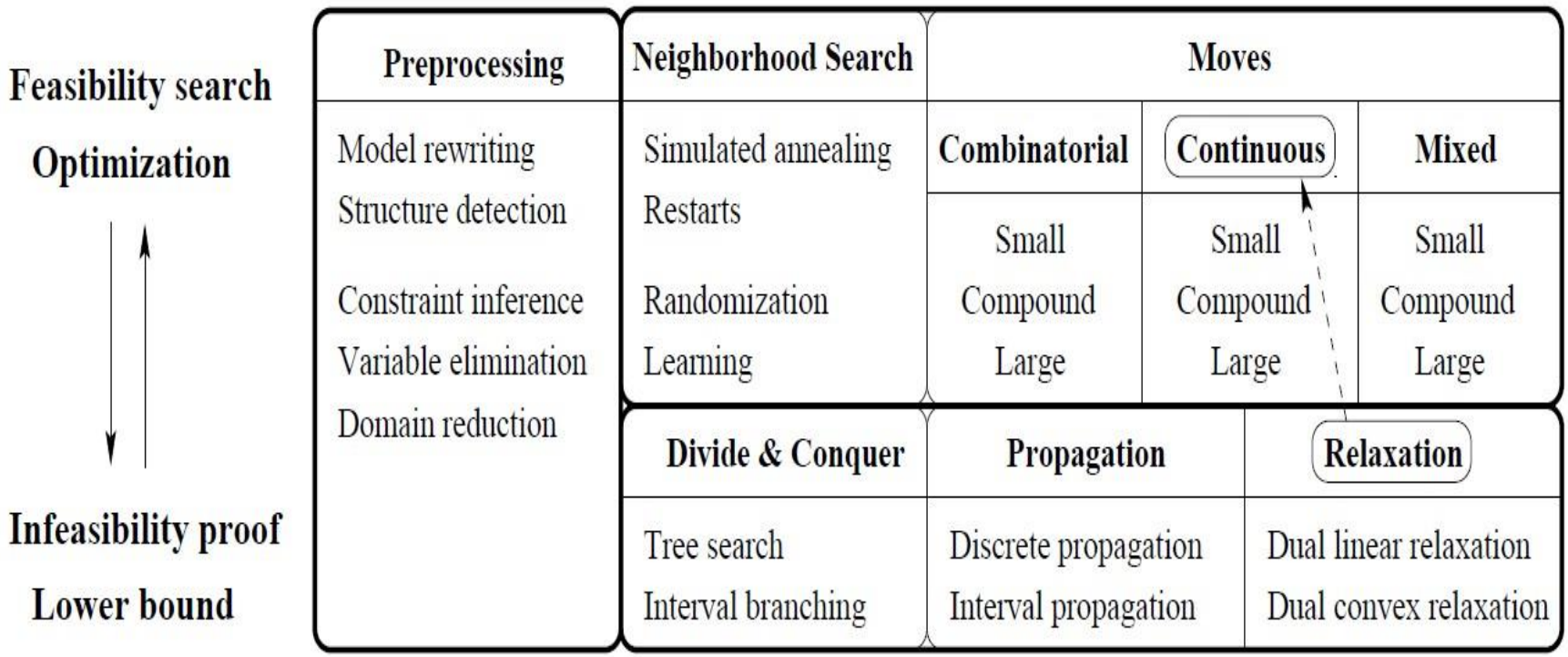
- 最少化問題
- 実行CPU時間：5分、
- PC： Intel Core i7-820QM (4 cores, 1.73 GHz, 6 GB RAM, 8 MB cache)

１．次世代の最適化を目指すLocalSolver8.0

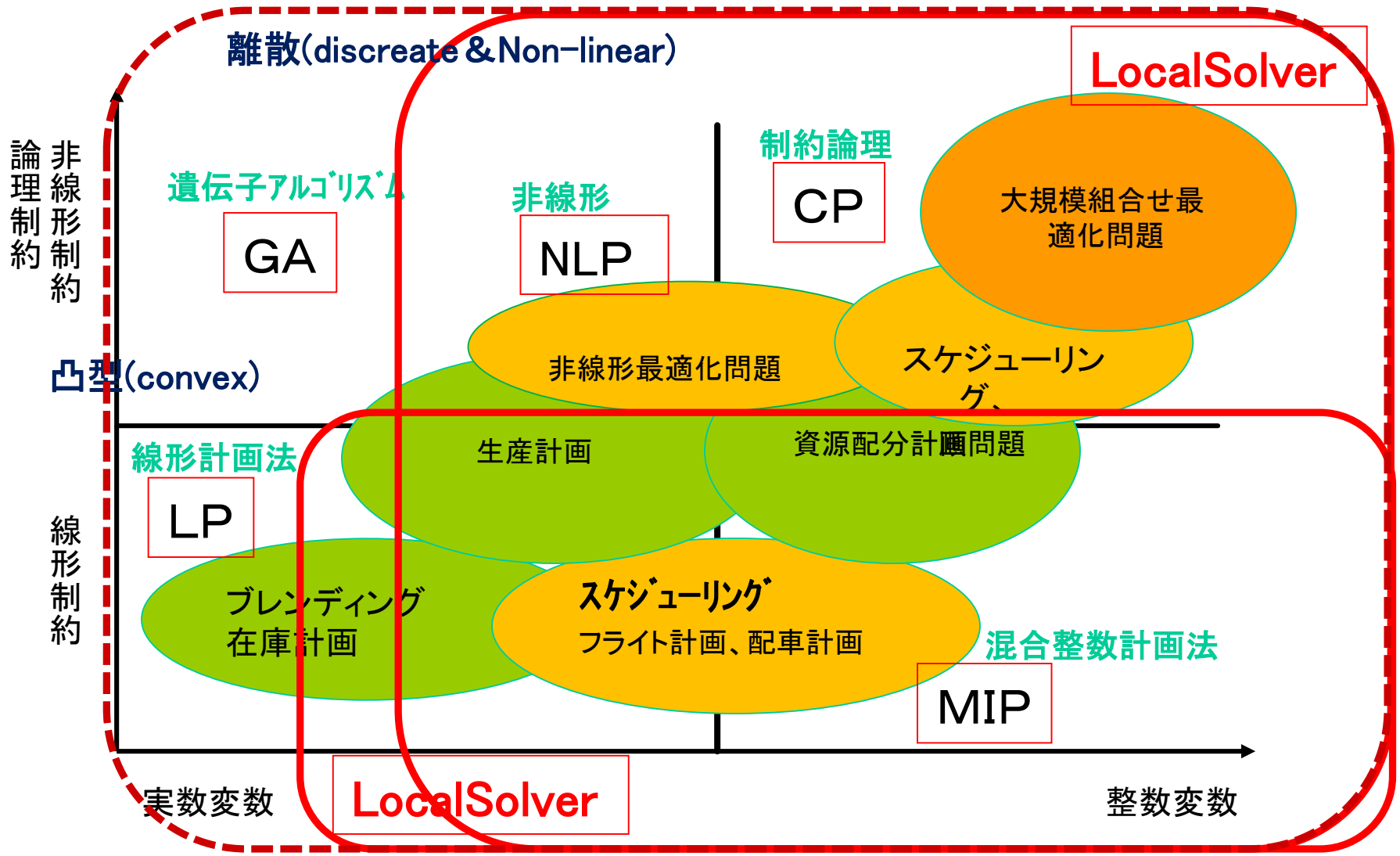
- グローバル探索を実現した大規模問題用の
  汎用数理計画法システム（All-In-One Solver）
- 様々な解法（メタ解法技術、既存解法技術）を融合
- 従来の解法依存型での定式化は必要なく、非線形問題をも含む汎用問題として定式化が可能
- 誰でも容易に問題定義ができ、使えることを実現

LocalSolver(様々なアーキテクチャを統合)

大規模混合整数変数、非凸最適化問題に対しても、
既存の最適化技術(LS, LP/MIP, CP/SAT, NLP, ...)を統合した**All-In-One**の数理計画法システムを実現。



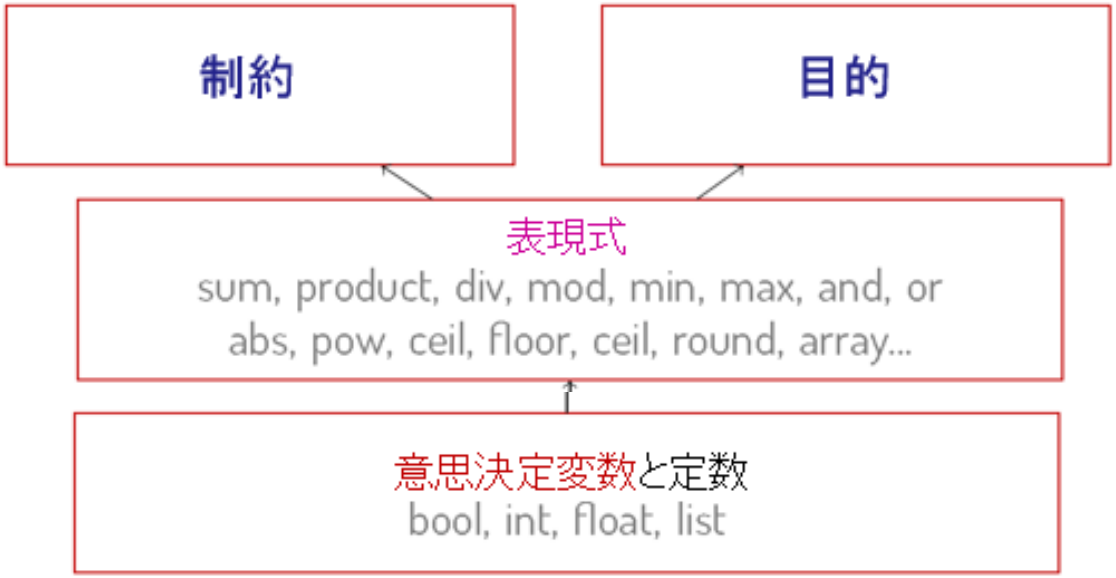
―LocalSolverの適用範囲―



最適化問題(組合せ最適化)の例

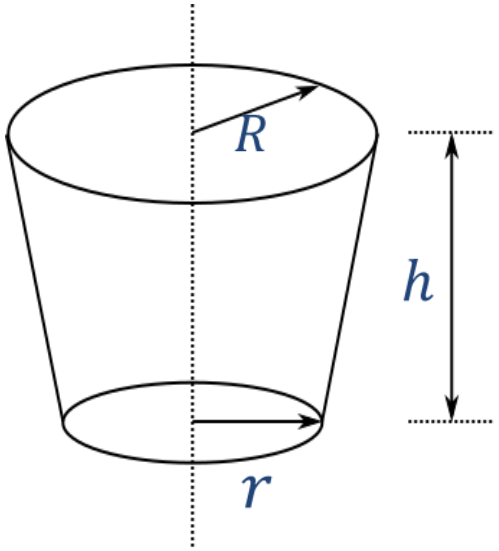
問題を自然な形で定義し実行するだけ

```
function model() {
    x[i in 0..nbItems-1] <- bool();
    knapsackWeight <- sum[i in 0..nbItems-1](weights[i] * x[i]);
    constraint knapsackWeight <= knapsackBound;
    knapsackValue <- sum[i in 0..nbItems-1](prices[i] * x[i]);
    maximize knapsackValue;
}
```



非線形最適化問題の例

体積が最大になるような形状の設計



$$V = \frac{\pi h}{3}(R^2 + Rr + r^2)$$

$$S = \pi r^2 + \pi(R + r)\sqrt{(R - r)^2 + h^2}$$

```
function model() {

  R <- float(0,1);
  r <- float(0,1);
  h <- float(0,1);

  V <- PI * h / 3.0 * (R*R + R*r + r*r);
  S <- PI * r * r + PI*(R+r) * sqrt(pow(R-r,2) +
h*h);

  constraint S <= 1;
  maximize V;
}
```

目的関数、制約条件を表現するオペレータの例

意思決定変数	算術型			論理型	大小比較	Set関係
bool	sum	sub	prod	not	eq	count
float	min	max	abs	and	neq	indexof
int	div	mod	sqrt	or	geq	partition
list	log	exp	pow	xor	leq	disjoint
	cos	sin	tan	iif	gt	
	floor	ceil	round	array+at	lt	
	dist	scalar		piecewise		

意思決定変数を定義し、上記オペレータを使って、

目的関数、制約条件を表現するだけでモデリングが完成。

後は実行させるだけ。

LocalSolver（技術：ノウハウの集大成）

- 解空間の徹底的な解析
  - **最新の問題解析による上界の計算（凸最適化問題）**

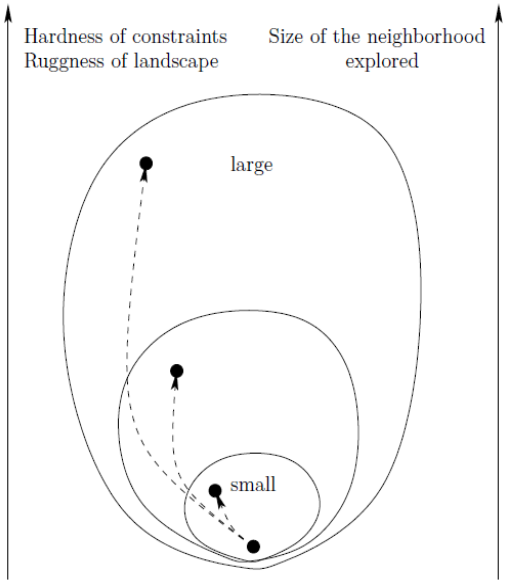
- 拡張した局所探索法をベース
  - **ハイパーグラフ理論を活用した汎用的な探索を実現**
  - 非凸、非平滑な解空間でも最適化計算が可能
  - 解を改良していく仕組み（解の評価と選択）：**百万回の探索/分**
  - 自律的にシミュレーティドアニーリングを実施

- C++で効率的に開発（IT技術の徹底的活用）
  - モデルの縮小と再定式化をする事前処理
  - **マルチスレッドを活用した探索**
  - 高度に最適化したメモリ管理

LocalSolver（グローバル探索イメージ）

グローバル探索としての近傍探索

- 小さい近傍から探索を広げていく
- 近傍探索からダイナミックに探索領域を調整：縮小、拡大、特化
- 大規模領域探索にツリー探索(MIP、CP)を導入
- 完全な近傍と正確な探索では最適解に到達

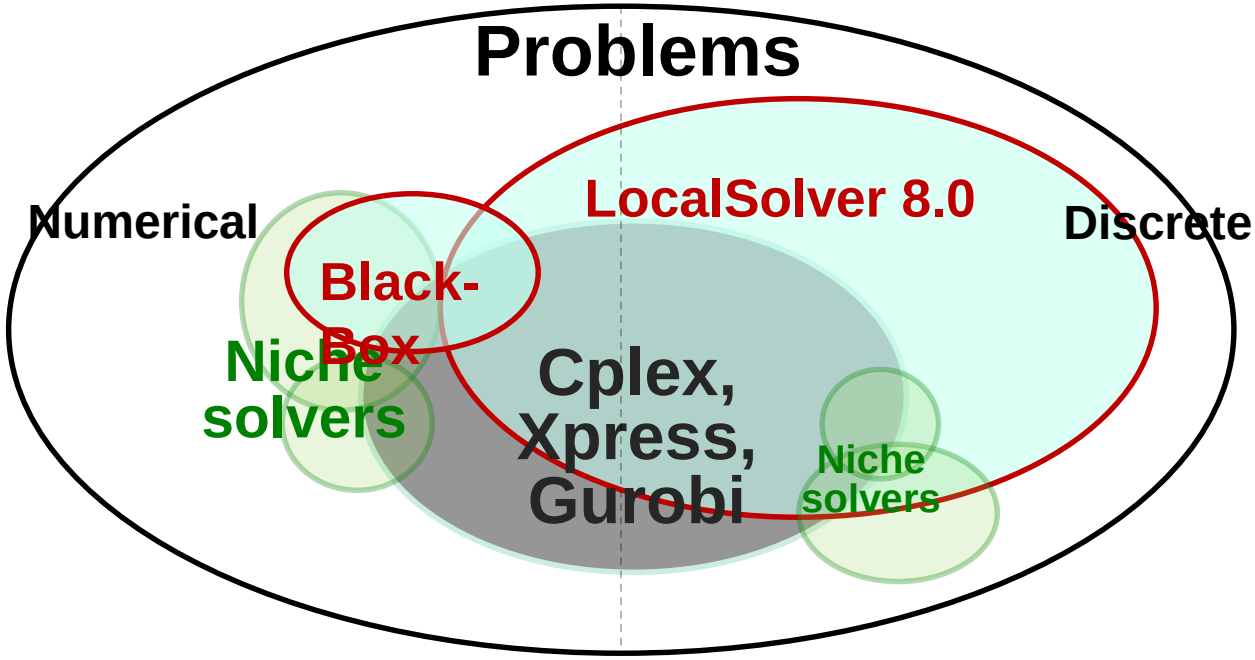


LocalSolverのAPI ：

- 独自の関数型言語LSPによる開発、実行環境
- C++、C#、JAVA、の他に、Python 2.7、3.2、3.4、
- 3.6 用のAPIを装備.
- GAMS モデリング言語からの使用（www.gams.com 参照）
- AMPLからの利用
- Pythonからの利用

2. LocalSolver8.0の新機能

- 主な機能
  - Integration of the power of LP/MIP techniques into LocalSolver
  - Development of set-based modeling features and solving performance



2. LocalSolver8.0の新機能（1）

上界の計算（最適性の証明）

より多くの問題に対する下限と最適性を証明する。

ブール値、整数または連続変数で線形または非線形で表現されたモデルの場合、LocalSolverはより良い下限を出力する。

特に2次モデルの場合。

実際には、線形リラクゼーション（MIPで使用されるリラクゼーション）非線形モデルに拘束力を与えることができる凸状/半限定的な弛緩も含む。

TOYモデル（最大化）

- C:¥localsolver_8_0¥examples¥toy>localsolver toy.lsp
- LocalSolver 8.0 (Win64, build 20180802)
- Copyright (C) 2018 Innovation 24, Aix-Marseille University, CNRS.
- All rights reserved. See LocalSolver Terms and Conditions for details.
- Load toy.lsp...
- Run model...
- Initialize preprocessing...
- Preprocess model 100% ...
- Close model 100% ...
- Run param...
- Run solver...
- Initialize threads 100% ...
- Push initial solutions 100% ...
- Model:
- expressions = 38, operands = 50
- decisions = 8 (bool = 8, int = 0, float = 0, list = 0, set = 0),
- constraints = 1, objectives = 1, constants = 11
- Param:
- time limit = 1 sec, no iteration limit
- seed = 0, nb threads = 2, annealing level = 1
- Objectives:
- **Obj 0: maximize, bound = 331**
- Phases:
- Phase 0: time limit = 1 sec, no iteration limit, optimized objective = 0
-
- Phase 0:
- [0 sec, 0 itr] : obj = 0
- [0 sec, 28383 itr] : obj = 280
-
- 28383 iterations, 56249 moves performed in 0 seconds
- Optimal solution: obj = 280
- **Upper bound : ub = 280 (gap < 0.01%)**
-
- C:¥localsolver_8_0¥examples¥toy>

TSPモデル(最小化)

```
C:¥localsolver_8_0¥examples¥tsp>localsolver tsp.lsp inFileName=instances/br17.atsp
LocalSolver 8.0 (Win64, build 20180802)
Copyright (C) 2018 Innovation 24, Aix-Marseille University, CNRS.
All rights reserved. See LocalSolver Terms and Conditions for details.
Load tsp.lsp...
Run input...
Run model...
Initialize preprocessing...
Preprocess model 100% ...
Close model 100% ...
Run param...
Run solver...
Initialize threads 100% ...
Push initial solutions 100% ...
Model:
  expressions = 50, operands = 334
  decisions = 1 (bool = 0, int = 0, float = 0, list = 1, set = 0),
  constraints = 1, objectives = 1, constants  = 17
Param:
  time limit = 5 sec, no iteration limit
  seed = 0, nb threads = 2, annealing level = 1
Objectives:
  Obj 0: minimize, bound = 0
Phases:
  Phase 0: time limit = 5 sec, no iteration limit, optimized objective = 0
Phase 0:
[  0 sec,       0 itr] : infeas = 8 violated expressions
[  1 sec,   12136 itr] : obj =          39
:
[  5 sec,  608767 itr] : obj =          39
608767 iterations, 1217616 moves performed in 5 seconds
Feasible solution: obj =          39
Lower bound     :  lb =           0 (gap = 100.00%)
Run output...

C:¥localsolver_8_0¥examples¥tsp>
```

PMEDIAN（非線形問題）

- C:¥localsolver_8_0¥examples¥pmedian>localsolver pmedian.lsp nFileName=instances
- /pmed1.in
- LocalSolver 8.0 (Win64, build 20180802)
- Copyright (C) 2018 Innovation 24, Aix-Marseille University, CNRS.
- All rights reserved. See LocalSolver Terms and Conditions for details.
- Load pmedian.lsp...
- Run input...
- Run model...
- Initialize preprocessing...
- Preprocess model 100% ...
- Close model 100% ...
- Run param...
- Run solver...
- Initialize threads 100% ...
- Push initial solutions 100% ...
- Model:
- expressions = 10489, operands = 40202
- decisions = 100 (bool = 100, int = 0, float = 0, list = 0, set = 0),
- constraints = 1, objectives = 1, constants = 286
- Param:
- time limit = 10 sec, no iteration limit
- seed = 0, nb threads = 2, annealing level = 1
- Objectives:
- **Obj 0: minimize, bound = 0**
- Phases:
- Phase 0: time limit = 10 sec, no iteration limit, optimized objective = 0
- hase 0:
- [0 sec, 0 itr] : obj = 59800
- :
- [10 sec, 99485 itr] : obj = 5819
- [10 sec, 99485 itr] : obj = 5819
- 99485 iterations, 197497 moves performed in 10 seconds
- Feasible solution: obj = 5819
- **Lower bound : lb = 0 (gap = 100.00%)**
- Run output..
- C:¥localsolver_8_0¥examples¥pmedian.

2. LocalSolver8.0の新機能（2）

リスト変数とセット変数の強化（大規模組み合わせ問題対応）による定式化と最適化

リスト変数とセット変数によるセットベースのモデリング機能をフルに利用できるようにした（LocalSolver6．5から大幅に性能強化）。

ブール変数を使用して表現されるパッキングの問題に役立つ。

これらのすべてのセットベースのモデリング機能では、パフォーマンスの向上を実現した。

特にルーティングおよびスケジューリングの問題：車両のルーティング、ピックアップおよび配送フローショップ、ジョブショップなど、すべてのルーティングとスケジューリングの問題に対して、LocalSolverは市場で最も優れたソルバーとなった。

Set-based modeling(List意思決定変数セットとは)

Structured decisional operator **list(n)**

- listセットは {0, …, n-1} の値のサブセットである。
- 一つのセット内は異なる値を持つ。

List(n)に対するオペレータ:

- **count(u)** : number of values selected in the list
- **get(u,i) or u[i]** : value at index i in the list
- **indexOf(u,v)** : index of value v in the list
- **contains(u,v)** : “indexOf(u,v)!= -1” と同じ意味
- **disjoint(u1, u2, …, uk)** : list を切り離す
- **partition(u1, u2, …, uk)**:listを一つの物として見る

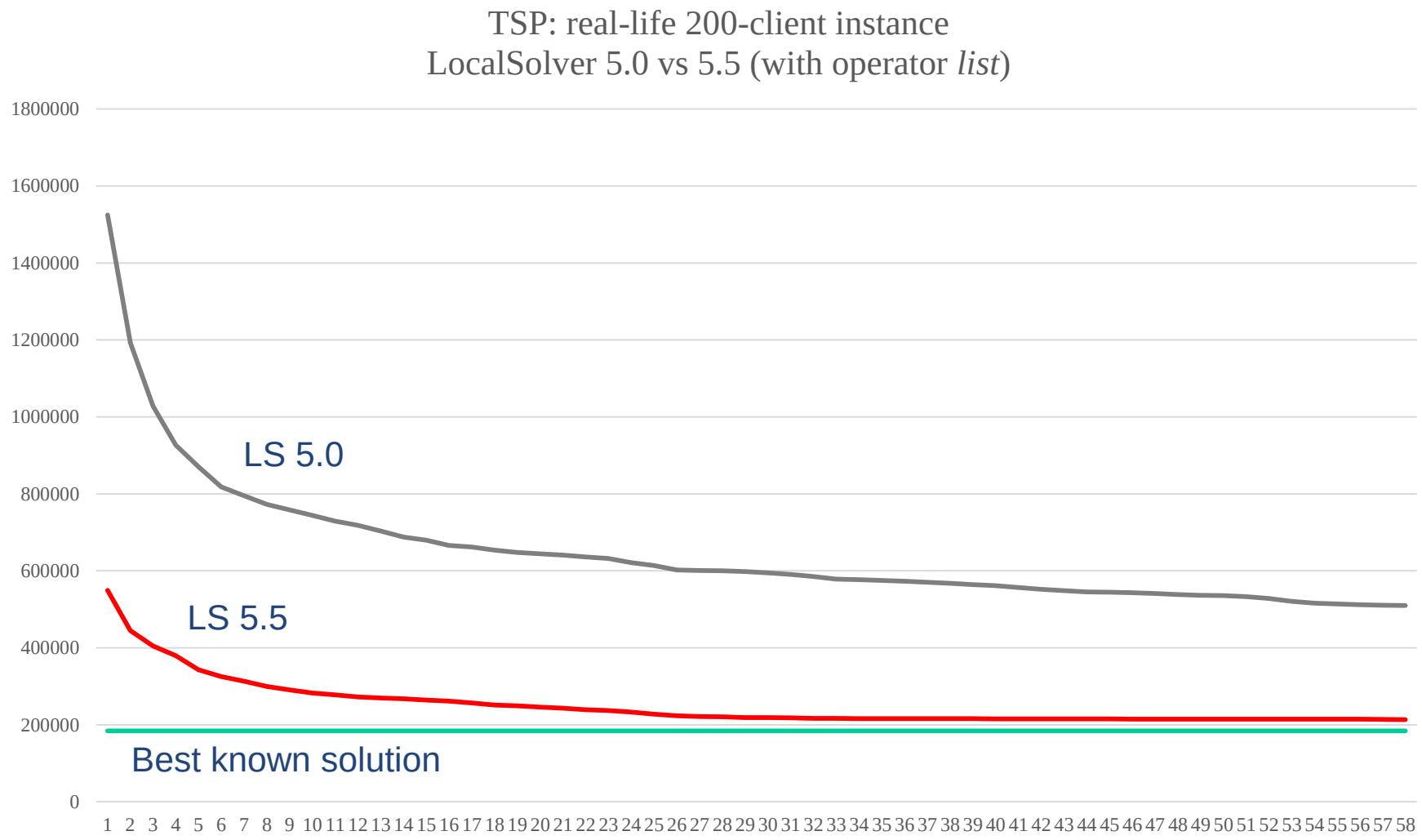
List を使ったTSPの定式化

```
function model() {
  x <- list(N) ; // order n cities {0, ..., n-1} to visit
  constraint count(x) == N; // exactly n cities to visit
  minimize sum[i in 1..N-1]( Dist[ x[i-1] ][ x[i] ] )
                    + Dist[ x[N-1] ][ x[0] ] ; // minimize sum of traveled distances
}
```

Could you imagine simpler model?

- Natural declarative model: straightforward to understand
- Common set-oriented concepts: easy to learn
- Still easier for people with (basic) programming skills
- Compact: linear in the size of input → highly scalable (1 million nodes)

List を使ったTSPの性能



List を使ったVRPの定式化

```
function model() {
  x[1..K] <- list(N) ; // for each truck, order the clients to visit
  constraint partition( x[1..K] ); // each client is visited once
  distances[k in 1..K] <- sum[i in 1..N-1]( dist( x[k][i-1], x[k][i]) )
            + dist( x[k][N-1], x[k][0] ); // traveled distance for each truck

  minimize sum[k in 1..K]( distances[k] ); // minimize total traveled distance
}
```

To go further, to make it simpler

- Sets (unordered) versus lists (ordered)
- Multi-sets/lists: multiple occurrence of the same values
- Collections of objects instead of values
- Ability to iterate and project over collections (lambda expressions)

3. 日本の事例

3-1 SCMモデル:超大規模モデル

1. 問題概要

顧客要求に合わせて、予め決められた工場ー顧客間の配送便に間に合うように、いつどこ向けになにを生産するかを決める問題（複数工場の生産スケジュール）

2. モデル概要

ー種類の0-1整数の意思決定変数（Bool変数）で、すべての制約を表現した。また、複数の目的関数を設定し、現実的な解法を実現した。

SCMモデルの実行結果比較

実行結果（1） 既存システム

全国規模を一つのモデルにすると、実用時間内で解くことができなかった（実行CPU時間を5分程度にするのに、全国規模のモデルを **20分割**にする必要があった)。

実行結果（2） LocalSolver

LocalSolverでは、**5～6分で全国規模**を一つのモデルで解くことができた。

- Bool変数: **800万以上**
- 複数の目的関数を重要な順番で設定し、順番に最適化計算を実行。

SCMモデルの実行結果

Phase 0:
[0 sec, 0 itr]: obj = (75593, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0), mov = 0, inf < 0.1%, acc < 0.1%, imp = 0

[1 sec, 27226 itr]: obj = (64967, 0, 6534, 0, 0, 0, 29280, 2838.45, 0, 519, 1059, 28.5621, 10132.2), mov = 48265, inf = 55%, acc = 45%, imp = 17073

[2 sec, 65843 itr]: obj = (46593, 0, 23231, 0, 0, 0, 39820, 3763.47, 0, 1287, 1145, 30.6949, 27794.2), mov = 124782, inf = 50.3%, acc = 49.6%, imp = 51706

[7 sec, 237135 itr]: obj = (11, 0, 51402, 0, 22, 0, 46130, 4122.05, 0, 2702, 1161, 31.1616, 73535.1), mov = 477135, inf = 60.2%, acc = 38.9%, imp = 146374

[8 sec, 280000 itr]: obj = (11, 0, 51402, 0, 22, 0, 46130, 4122.05, 0, 2702, 1161, 31.1616, 73535.1), mov = 560000, inf = 64.6%, acc = 34.4%, imp = 146374

[9 sec, 320000 itr]: obj = (10, 0, 52006, 0, 25, 0, 46150, 4125.65, 0, 2713, 1160, 31.1471, 73522.5), mov = 640000, inf = 67.9%, acc = 31.1%, imp = 146375

[10 sec, 360000 itr]: obj = (10, 0, 52006, 0, 25, 0, 46150, 4125.65, 0, 2713, 1160, 31.1471, 73522.5), mov = 720000, inf = 70.4%, acc = 28.5%, imp = 146376
360000 iterations, 720000 moves performed in 10 seconds
Feasible solution: obj = (10, 0, 52006, 0, 25, 0, 46150, 4125.65, 0, 2713, 1160, 31.1471, 73522.5)

3-2 人員配置(アルバイト配置計画)

1. 意志決定変数の定義(Bool変数)

X(人員、日付、時刻、業務) ＝ {0, 1}

ここまでは装置のスケジューリングとあまり変わらないが・・・

2. 人のスケジューリング特有の制約

・最大勤務時間

・最小勤務時間←**あまり短いとアルバイトは出てこない**

・１日１シフト←**勤務の途中に何もしない時間を入れない**

を導入。

人員配置計画の実行結果比較

実行結果（1） 小モデル

人員：20名、1週間、1時間毎の16タイムスロット、3業務

(意志決定変数： 20 x 7 x 16 x 3 = 9520)

LocalSolver 3.1 ：最適解まで約20秒

既存MIP ：最適解まで約30秒

実行結果（2） 業務モデル

人員：83名、1週間、15分毎の54タイムスロット、5業務

(意志決定変数： 83 x 7 x 54 x 5 = 156870)

LocalSolver3.1：最適解まで約210秒

既存MIP ：600秒以上

（制約内容等は異なっているが・・・）

3-3 裁断計画

5000mmの幅をもつフィルム等のロールから、色々な幅を持つ複数製品のロールを要求本数にできるだけ近くなるよう、裁断パターンとパターンの使用回数を求める問題である。

目的関数（複合目的関数）

1. パターン数最小（＊1000）
2. 要求数量とのギャップ最小（＊100）
3. ロス部分(4600mmからの差)。

Valid Roll width		3600 ～		4600
Products	width	Request	minimum	maximum
1	600	10	8	12
2	740	15	12	18
3	920	20	16	24
4	1060	5	4	6
5	1200	10	8	12

Pattern No	1	2	3	4	5	6	16	17	18	19	26	27	28	51	52	
Product	volume count in pattern(i)							9本				2本				
600	0	0	1	0	0	0	1	0	1	0	1	4	0	6	7	8本
740	0	1	0	0	0	0	1	2	1	3	1	0	1	1	0	18本
920	1	0	0	0	1	2	1	2	2	1	1	0	3	0	0	18本
1060	0	0	0	2	1	0	1	0	0	0	2	2	1	0	0	8本
1200	3	3	3	2	2	2	1	1	1	1	0	0	0	0	0	9本
Total	4520	4340	4200	4520	4380	4240	4520	4520	4380	4340	4380	4520	4560	4340	4200	
rest	80	260	400	80	220	360	80	80	220	260	220	80	40	260	400	

裁断計画実行結果比較

実行結果（1） 通常モデル

製品種類数：10、総製品数量：153、切断パターン候補数：12898

(意志決定変数： 75302)

LocalSolver 3.1 ：11秒で4パターン、要求差＝3

既存MIP ：400秒すぎても9パターン、要求差＝10

実行結果（2） 大規模モデル

製品種類数：18、総製品数量：504、切断パターン候補数：70921

(意志決定変数： 399372)

LocalSolver 3.1 ：31秒で8パターン、要求差＝17

既存MIP ：4060秒で10パターン、要求差＝22

3-4　フランスの事例（Car sequencing）

車両投入計画（塗装及びアセンブル）

- 従来はラインに沿って部品を配置する計画
- 同一色塗装の連続制約
- 二次的目的として色の変更を最小限にしたい

大規模な最適化問題

- 1300台の投入順を決める → 400 000 バイナリ意志決定変数が必要
- MIP or CP ソルバーでは数時間の実行でも実行可能解さえ見つからない
- LocalSolver　は**数秒**で実用的な精度を持つ解を Renault　に提供

Car Sequencing at Renault（比較結果）

色に関する自動車配列問題の結果を示す。LP緩和解（上限は3001（万））。

LocalSolver V3.0（式数：80850, オペランド数：295212、意志決定変数：44184, 制約条件：636, 目的関数：3）

- **1秒で目的関数：16122**の解を見つける
- 6秒で3478に達する
- **1分後に、3125**に達する（この例で最も有名なものは3109である）

Guroubi V5.1（式数: 29845, 列数：57331, 非零要素数：442093、連続変数：15777, バイナリ変数：41554）

- LP緩和解を求めるのに、87561イタレーション, 312 秒かかる
- B&Bで最初の実行可能解をみつけるのに**647秒**かかる
- 1766秒後に解27720を見つける
- **60分後でも**コストはまだ25000を越えている

大規模組合せ最適化問題に関しては、Branch&Boundに基づく探索では、探索が本質的に制限されるため、解くのが難しいのが実体である。
たとえヒューリスティクティクサーチ等の発見的探索法を使用するとしても限界がある。
※かなり良い線形緩和の問題（従来のMIP用ベンチマークデータ）では、従来型でもある程度実用的な時間で解を見つけることは可能である。

4. おわりに

　30年前には殆ど実現出来なかった大規模組合せ最適化問題に対して、実践的なアプローチが実現できる時代になったと考える。
　また、大規模最適化問題の定式化技術が進歩し、簡単に大規模問題の作成が可能となっている。

「実学に役立つOR」として、人間と機械の調和を実現して日本の産業界の再生の一助となれば幸いである。

参照文献

(1)T. Benoist, B. Estellon, F. Gardi, R. Megel, K. Nouioua (2011).

・「LocalSolver 1.x: a black-box local-search solver for 0-1 programming」、*4OR, A Quarterly Journal of Operations Research* 9(3), pp. 299-316. Springer.

(2)MSI株式会社

・「http://msi-jp.com/localsolver/」ホームページ

(3) Frédéric Gardi(2013)

・「Habilitation Thesis in Computer Science:Toward a mathematical programming solver based on local search」、Université Pierre et Marie Curie (Paris 6) - Faculté d'Ingénierie (UFR 919) École Doctorale Informatique, Télécommunications et Électronique de Paris (EDITE).

(4) Frédéric Gardi(2013)

・「Toward a mathematical programming solver based on local search」、13th December 2013 ORO Workshop, Nantes.

宮崎　知明　MSI株式会社　技術顧問。数理モデリング研究所所長。

1．得意分野

（1）SCM：サプライチェーンマネージメント
- ・戦略レベルの最適化計画の策定およびビジネスモデルの提案
- ・戦術レベルの最適化計画の策定およびビジネスモデルの提案
- ・オペレーションレベルの各種スケジューリングの策定
- ・生産計画、在庫計画、物流計画、販売計画の策定

（2）OR：オペレーションズリサーチ
- ・モデリング
- ・解法設計、開発
- ・最適化（数理計画法ほか）
- ・スケジューリング、計画支援

2．業務履歴

（1）1974年3月：東京大学工学部応用物理学科物理工学コース卒業
（1）1974年4月：富士通(株)に入社
（2）1974年－1983年：数理計画法の研究、製品開発及び顧客適用業務に従事
（3）1983年－1991年：OR手法全般の製品化戦略及び顧客適用業務に従事
（4）1992年－1997年：OR手法全般、DWH、OLAP、DM製品戦略及び顧客適用業務に従事
（5）1998年－2001年：SCMパッケージ適用コンサルティング、ソリューション開発
（6）2002年4月－2010年6月（株）富士通総研に出向。数理解析技術による各種最適化の適用コンサルティング
（7）2010年7月-2012年11月　株式会社メルキュールシステムに入社。数理モデリング研究所を社内で設立。最適化コンサルティング。
（8）2012年12月；MSI株式会社　技術顧問。数理モデリング研究所所長。最適化コンサルティング。

＜顧客への主なコンサルティングプロジェクト＞

①石油精製A社：生産計画(予算計画)モデルの構築及びシステム導入支援
②石油元売B社：原油購入計画モデルの構築およびシステム開発
③製鉄業C社：生産計画モデルの構築及びシステム導入支援
④衣料製造業D社：人員配置及び生産計画システム策定及び解法設計
⑤自動車部品製造業E社：合金生産計画システム策定及び解法設計
⑥自動車製造業F社：エンジン制御シミュレーションシステムコンサルティング
⑦鋳鉄製造業G社：月次生産計画コンサルティング及び解法設計
⑧製紙業H社：生産計画コンサルティング及び解法設計
⑨製紙業I社：OLAP活用コンサルティング
⑩石油化学J社：月次工場生産計画コンサルティング及び解法設計
⑪石油化学K社：全社生産計画コンサルティング及び解法設計
⑫総合化学L社：配車配送計画コンサルティング及び解法設計
⑬食品製造業M社：企業合併に伴う生産物流戦略コンサルティング
⑭石油元売N社：タンクローリ配車スケジューリングシステム解法設計
⑮自動車製造業O社：完成車配送スケジューリングシステム策定及び解法設計
⑯電機製造業P社：SCMパッケージ導入コンサル及びプロジェクトマネージメント
⑰食品製造業Q社：生産、物流（SCM）計画導入コンサルティング
⑱エネルギー製造業R社：保守要員計画導入コンサルティング　　その他

＜コンサルティング技法、ツール開発、製品企画など＞
- ・数理計画法システムの製品企画及びシステムの開発
- ・OLAPツール製品企画
- ・DM（データマイニング）製品企画
- ・シミュレーションシステム製品企画
- ・物流最適化システム企画、開発
- ・プロセス最適化システム企画、開発
- ・フィールドイノベーション活動支援（顧客業務改善、改革）

＜対外発表・その他活動＞
- ・OR学会編集委員1996－2000
- ・OR学会、情報処理学会発表1995ー
- ・PSLXコンソーシアム評議委員
- ・SCC日本支部ボードメンバ1999
- ・OR普及賞受賞（富士通ソフトウェア事業部として）
- ・スケジューリング学会理事2007
- ・**日本オペレーションズ・リサーチ学会フェロー**2008
- ・筑波大学講師（2012年上期）、中央大学講師(2014年下期）

有難う御座いました。

LocalSolverについてのお問い合わせ

MSI株式会社（日本配給元）
〒261-7102千葉市美浜区中瀬2-6 WBGマリブウエスト2階
Tel:043-297-8841 Fax:043-297-8836
Eメール：localsolver@msi-jp.com