HITACHI
Inspire the Next

OSSプランニングエンジン OptaPlannerを使ってみよう！

2015/08/07

株式会社日立ソリューションズ
三浦 健太郎

© Hitachi Solutions, Ltd. 2015. All rights reserved.

HITACHI
Inspire the Next

Contents

© Hitachi Solutions, Ltd. 2015. All rights reserved.

HITACHI
Inspire the Next

1. OptaPlannerの概要

© Hitachi Solutions, Ltd. 2015. All rights reserved.

1-1. OptaPlannerとは

HITACHI
Inspire the Next

OptaPlanner (オプタプランナー)とは**組合せに関する**
さまざまな問題の最適解を導くためのJavaライブラリです

- ✓JBossコミュニティで開発されているソフトウェアの一つ
- ✓オープンソース(Apache Software License 2.0)
- ✓100% Javaで実装
- ✓有償サポート版もあります

© Hitachi Solutions, Ltd. 2015. All rights reserved.

1-2. 組合せ最適化問題とは

HITACHI
Inspire the Next

いろいろな制約のもとで複数の選択肢の中から最善な組合せを求めること

（例）最短経路で京都の観光スポットを回るには…？
⇒ 観光スポットの訪問順の組合せ

- ✓対象数が増えるとパターンが劇的に増加する
- ✓コンピュータで全パターンを計算させると
  現実的な時間に終わらない

- ✓組合せ最適化は世界中で研究されており、より速く、より正確な解を出すための解法が日々研究されている
- ✓プログラミングコンテストのお題になっている

スポット数	パターン数	処理時間*1
3	6	1秒以内
4	24	1秒以内
5	120	1秒以内
:	:	:
10	10^6	42日
:	:	:
20	**10^{18}**	**7700万年**

*1）1000パターン/秒の処理ができると仮定した場合

© Hitachi Solutions, Ltd. 2015. All rights reserved.

1-3. 組合せ最適化問題の応用例

HITACHI
Inspire the Next

人が無意識に考えているものからビジネスまで いろいろなところで応用されています

◆身近な最適化

- ✓最短経路で京都の観光スポットを回るには…？
- ✓お小遣いを最も満足できるように使うには…？
- ✓どのタスクから着手したら最も成果が出るかな？

◆ビジネスでの最適化

- ✓データセンターにおけるサーバやリソース割当の最適化
- ✓従業員シフト勤務の最適化
- ✓配送スケジュール・配送ルートの最適化
- ✓生産計画の最適化
- ✓経営戦略の決定
- ✓財務の最適化

© Hitachi Solutions, Ltd. 2015. All rights reserved.

1-4. 最適化問題を解くためのソフトウェア

HITACHI
Inspire the Next

有償なものからオープンソースまで多々あるが
数式でモデル化するなど数学の知識が必要

◆有償ソフトウェアの例

- ✓ *Gurobi*
- ✓ *SCOP*
- ✓ *OptSeq*
- ✓ *CPLEX*
- ✓ *LocalSolver*

◆オープンソースの例

- ✓ *Scipy*
- ✓ *GLPK*
- ✓ *CBC*
- ✓ *lp_solve*
- ✓ ***OptaPlanner***

OptaPlannerはオブジェクト指向でモデル化できるため
数学の知識がなくても(とりあえず)使えます！

本セミナーでは学術的な用語をできるだけ使わずに
OptaPlannerの使い方を説明します

© Hitachi Solutions, Ltd. 2015. All rights reserved.

1-5. OptaPlannerを使うために必要な準備

HITACHI
Inspire the Next

◆モデル化

✓組合せ対象を決める
✓オブジェクトとして実装する

◆最適解の定義

✓どんな状態が最適解なのかを
スコア計算という形で実装する

◆設定ファイルの作成

✓探索アルゴリズムを設定する
✓終了条件を設定する

シフト作成の場合

◆モデル化

シフトと従業員を組合せたいので
それぞれのオブジェクトを実装する

8/1 Aシフト
8/1 Nシフト
8/2 Aシフト
：

：

◆最適解の定義

同じ日に複数のシフトを
割当てないこと

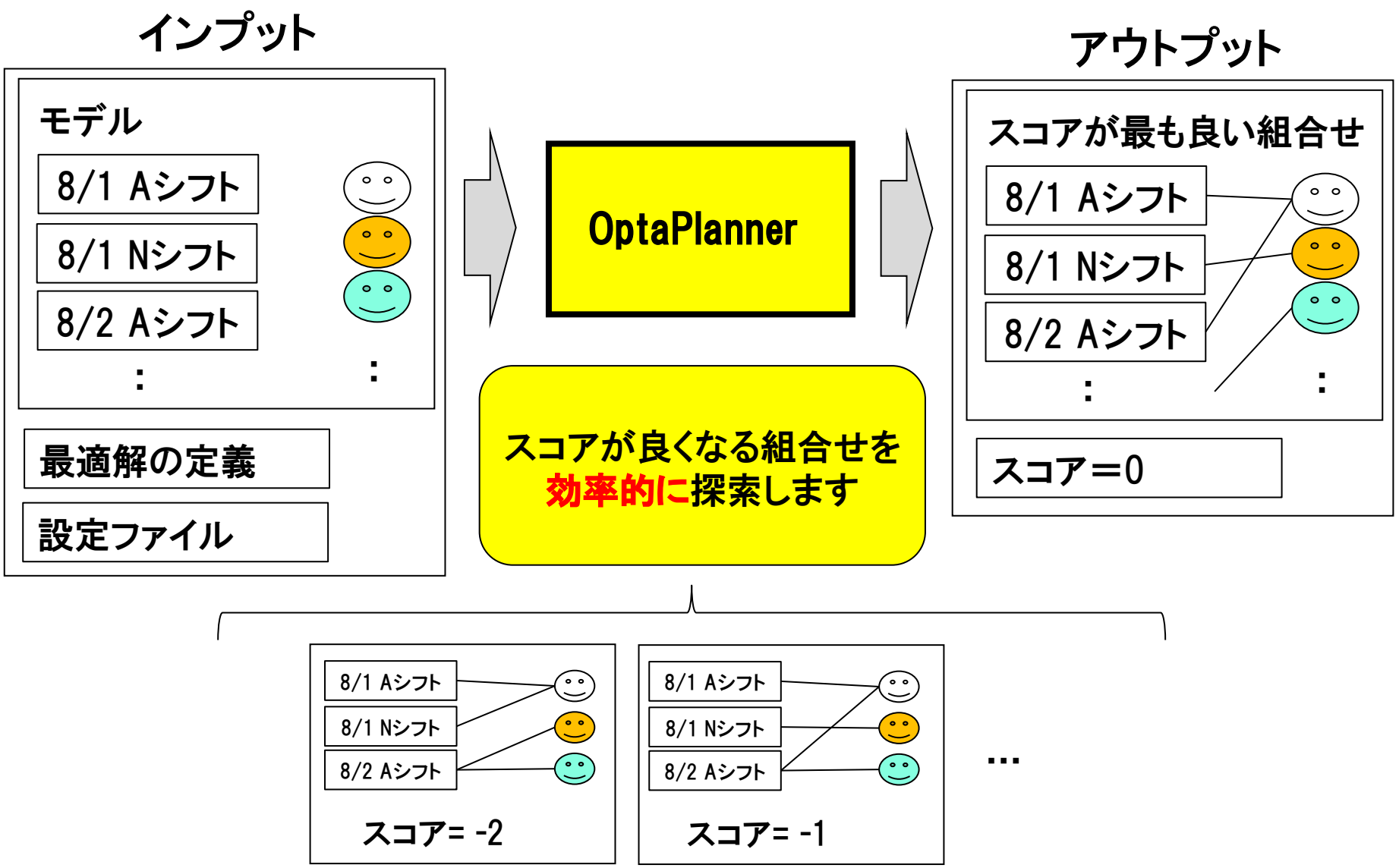
8/1 Aシフト
8/1 Nシフト

＝ スコア -1

© Hitachi Solutions, Ltd. 2015. All rights reserved.

1-6. OptaPlannerがやってくれること

HITACHI
Inspire the Next



© Hitachi Solutions, Ltd. 2015. All rights reserved.

1-7. 効率的に探索する仕組み

HITACHI
Inspire the Next

メタヒューリスティックという手法を用いている

✓最適化問題を解くための経験的(人間くさい)手法を結合させたもの
✓短時間で高精度な近似解を得ることができる
✓得られる解に理論的な保証はない

ベースになっているのは局所探索法(local search)

✓値を変更したり(change)、入替えたり(swap)して組合せを少しずつ変えて、
　より良い解を探すアルゴリズム
✓値の変更の仕方にバリエーションがあり、いくつかのアルゴリズムが
　ビルトインされている
✓探索方法をカスタマイズできる

© Hitachi Solutions, Ltd. 2015. All rights reserved.

1-8. OptaPlannerの特徴1

HITACHI
Inspire the Next

◆フレームワーク化されているので

✓組合せ問題を簡単にモデル化できます
✓実装済みのさまざまな探索アルゴリズムを利用できます
✓探索アルゴリズムのカスタマイズが容易にできます

◆Javaライブラリなので

✓既存資産を流用できます
✓どのプラットフォームでも動きます
✓他のJavaテクノロジーとの連携が容易です

◆オープンソースなので

✓無償で利用できます(Apache Software License 2.0)
✓有償サポートもあります(Red Hat JBoss BRMSに含まれます)

© Hitachi Solutions, Ltd. 2015. All rights reserved.

1-9. OptaPlannerの特徴2

HITACHI
Inspire the Next

◆学術的な知識がなくても使えます

効率的に探索できているかを評価するためのベンチワーク機能があります。
複数の探索アルゴリズムの実行結果を比較できるレポートを作成してくれ、
どの探索アルゴリズムが適しているかを分析できます。

◆直観的にモデル化できます

✓オブジェクト指向でモデル化できます
✓スコア計算をDRLで実装できます

DRL(Drools Rule Language)とは
Drools(ルールエンジン)で使用されるルール言語。

© Hitachi Solutions, Ltd. 2015. All rights reserved.

HITACHI
Inspire the Next

2. OptaPlannerのexampleの紹介

© Hitachi Solutions, Ltd. 2015. All rights reserved.

2-1. OptaPlannerのexampleの紹介

HITACHI
Inspire the Next

コミュニティサイトでexampleが提供されています
いろいろなモデルがあるので実装の参考にできます

- コミュニティサイトからリリース物(optaplanner-distribution-6.2.0.Final)をダウンロードします。
  http://www.optaplanner.org/
- 任意のディレクトリに解凍します。
- examples/runExamples.bat(Linux/Macの場合runExamples.sh)を実行します。

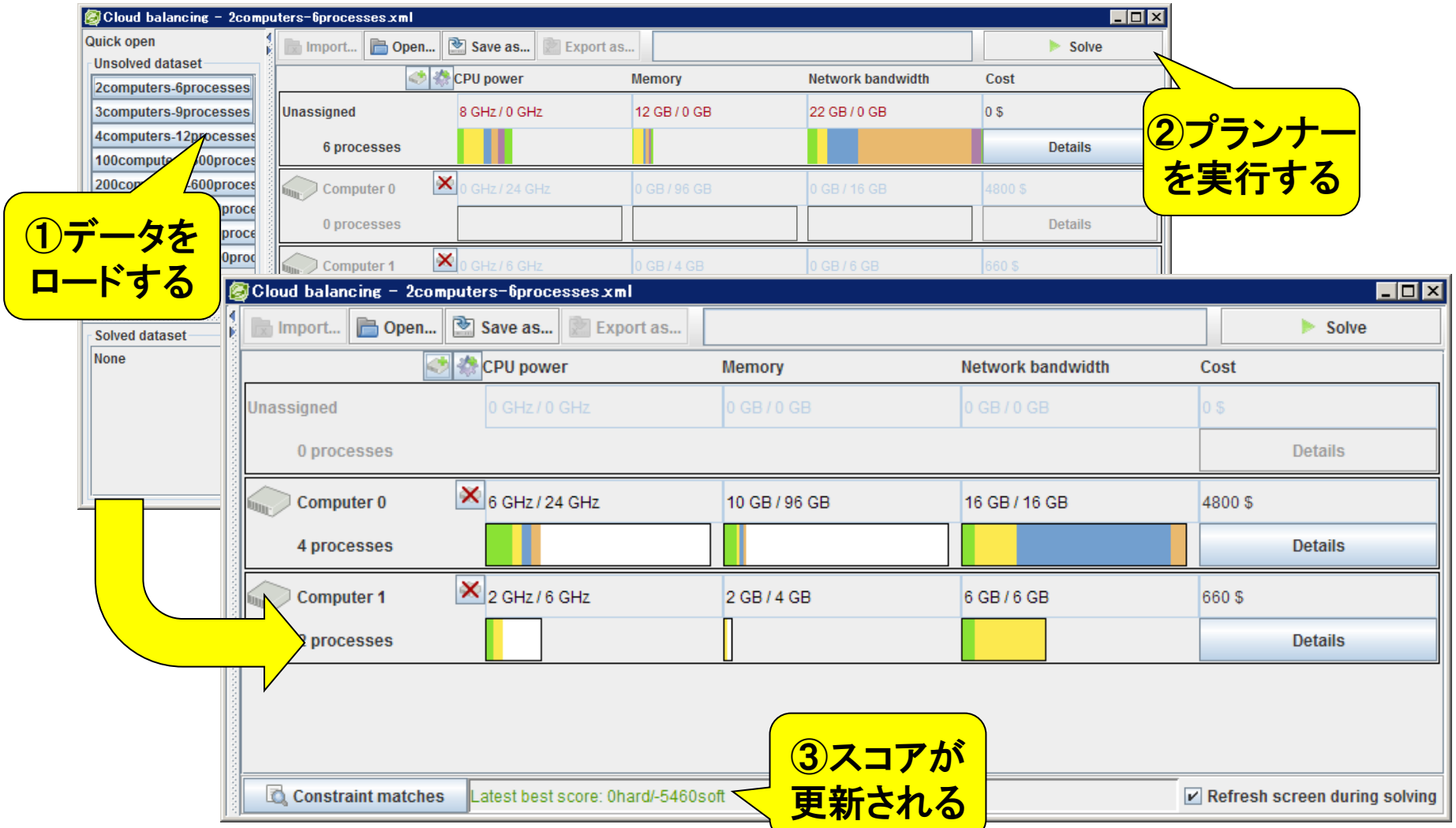
※実行にはJREまたはJDKが必要です。

© Hitachi Solutions, Ltd. 2015. All rights reserved.

2-2. Cloud balancing

HITACHI
Inspire the Next

- プロセスは実行に必要なCPU、メモリ、ネットワーク帯域を持っている。
- コンピュータのリソースを超えないようにプロセスを割当てる問題。

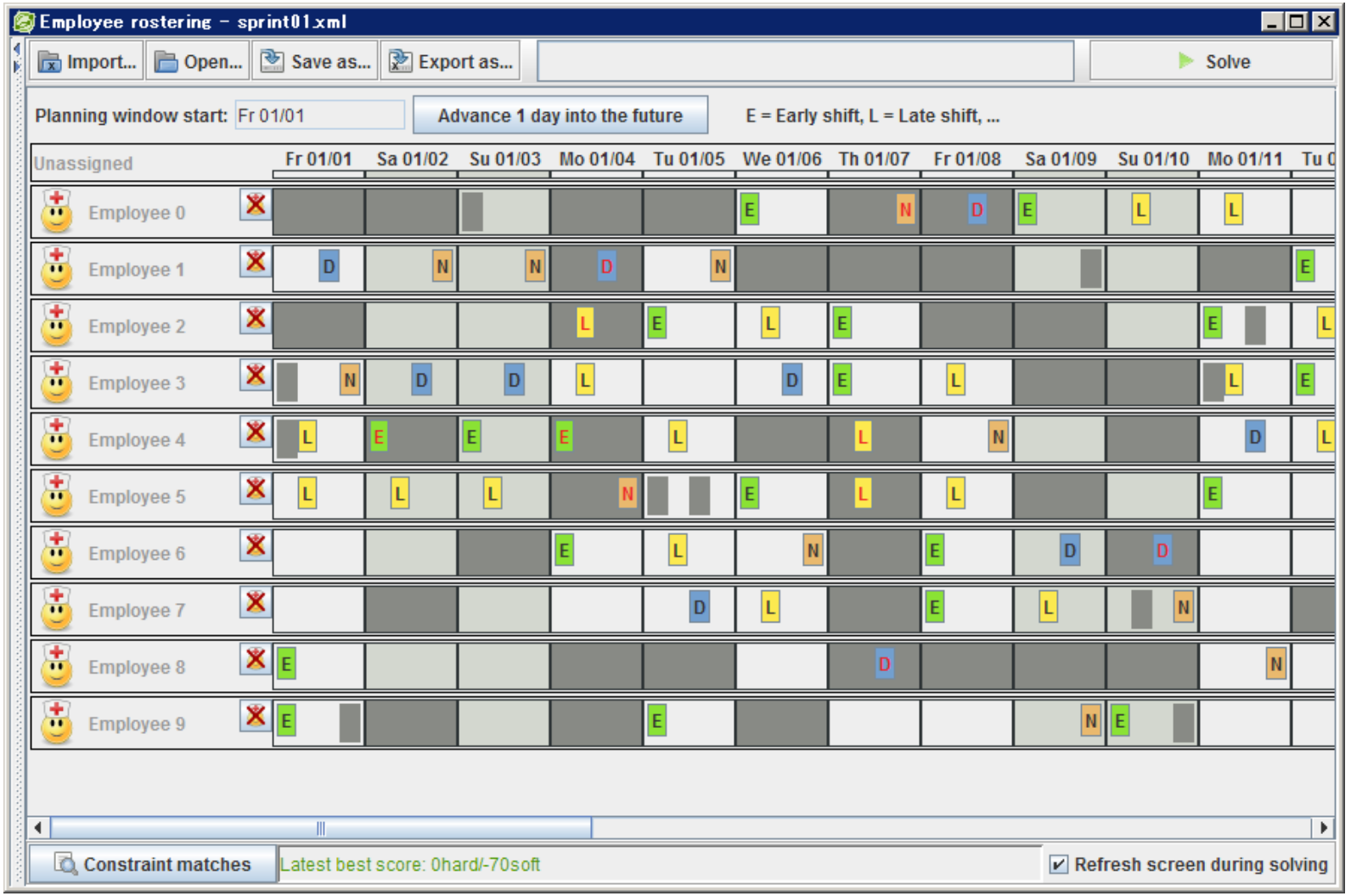


© Hitachi Solutions, Ltd. 2015. All rights reserved.

2-3. Employee rostering

HITACHI
Inspire the Next

- 従業員をシフトに割当てる問題。
- 各従業員の休日希望日や就業規則に違反しないようにしつつ、できるだけ公平にシフトを割当てる。

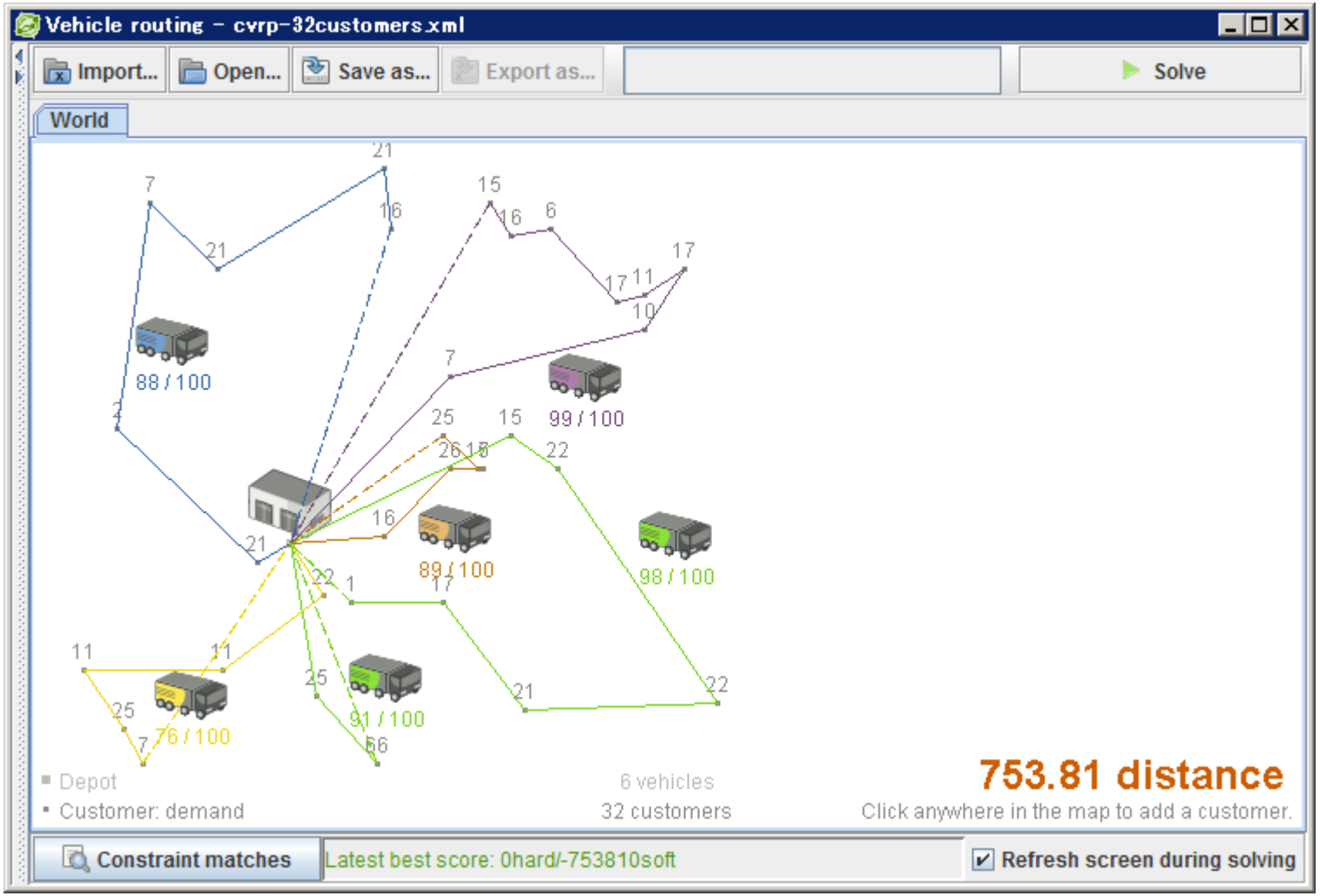


© Hitachi Solutions, Ltd. 2015. All rights reserved.

2-4. Vehicle routing

HITACHI
Inspire the Next

- トラック配送問題。各顧客の荷物をピックアップし、それを発着所に持っていく。
- 各トラックは複数の顧客にサービスできるが、容量は上限がある。



© Hitachi Solutions, Ltd. 2015. All rights reserved.

HITACHI
Inspire the Next

3. OptaPlannerの実装例

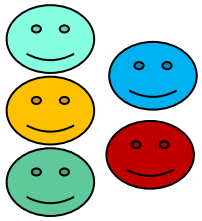
© Hitachi Solutions, Ltd. 2015. All rights reserved.

HITACHI
Inspire the Next

OptaPlannerを使ってシフト作成を自動化します

◆要件

✓シフト従業員は5人
✓シフトは「朝シフト」、「昼シフト」、「夜シフト」の3種類で
1日につき各シフトに1人の割当てが必要



	8/1	8/2	8/3	8/4	…
朝シフト					
昼シフト					
夜シフト					

◆最適解の定義

✓すべてのシフトに従業員が割当てられること
✓同一日に同じ従業員が複数のシフトに割当てられないこと
✓できるだけ各シフトが均等に割当てられること
✓できるだけ夜シフトの翌日に朝シフトが割当てられないこと

© Hitachi Solutions, Ltd. 2015. All rights reserved.

3-2. 実装の流れ

HITACHI
Inspire the Next

(1) モデル化する
(2) モデルを実装する
(3) 最適解を定義する
(4) スコア計算を実装する
(5) 探索アルゴリズムを決める
(6) 終了条件を決める
(7) 設定ファイルを作成する
(8) 実行する
(9) テスト・デバッグする

© Hitachi Solutions, Ltd. 2015. All rights reserved.

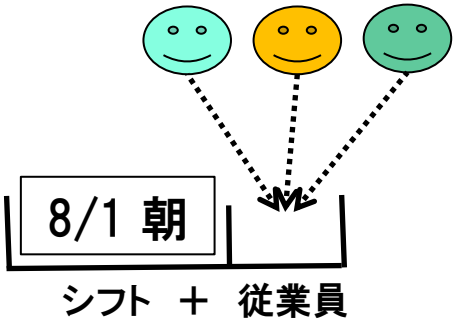
HITACHI
Inspire the Next

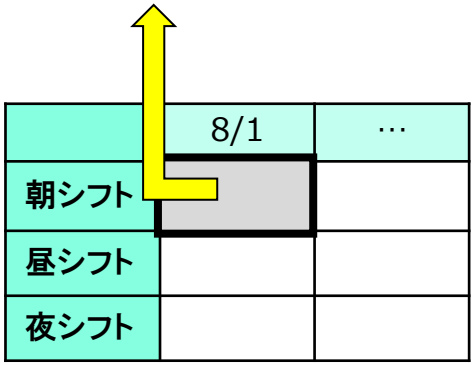
組合せ対象を抽出し、変数とそれを入れる箱を決める

◆シフト作成の設計の1例

変数 ＝ 従業員

箱 ＝ シフト ＋ 従業員 ＝ シフトアサイン



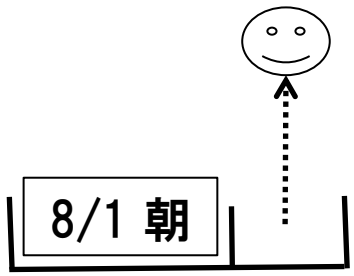


✓変数と箱はN:1の関係にする

© Hitachi Solutions, Ltd. 2015. All rights reserved.

HITACHI
Inspire the Next

クラス定義に変数と箱を示す印(アノテーション)を付ける




```
public class Employee {
    private String name;
    // getter and setter
}
```

```
@PlanningEntity
public class ShiftAssignment {
    private Shift shift
    private Employee employee;

    @PlanningVariable
    public Employee getEmployee() {
        return employee;
    }
    // getter and setter
}
```

- ✓変数クラスには何も付けない
- ✓箱クラスに*@PlanningEntity*を付ける
- ✓箱クラスの変数のgetterに*@PlanningVariable*をつける

© Hitachi Solutions, Ltd. 2015. All rights reserved.

HITACHI
Inspire the Next

変数と箱のオブジェクトを管理するクラスを実装する

```
@PlanningSolution
public class ShiftSolution implements Solution {
    private List<Employee> employeeList;
    @PlanningEntityCollectionProperty
    private List<ShiftAssignment> shiftAssignmentList;
    :
    public Collection<Object> getProblemFacts() {
        List<Object> facts = new ArrayList<Object>();
        facts.addAll(employeeList);
        return facts;
    }
```

Employeeオブジェクトを保持するリスト

ShiftAssignmentオブジェクトを保持するリスト

PlanningEntityオブジェクトの追加は不要
(OptaPlannerが挿入するため)

- ✓ *@PlanningSolution*を付ける
- ✓Solutionインタフェースを実装する
- ✓箱オブジェクトの管理変数に*@PlanningEntityCollectionProperty*を付ける
- ✓スコア計算をDRLで実装する場合はオブジェクトをワーキングメモリに挿入するためにgetProblemFacts()を実装する

© Hitachi Solutions, Ltd. 2015. All rights reserved.

HITACHI
Inspire the Next

最適解を制約とスコア計算で定義する

✓制約は複数のレベルに分けて定義できる(ここでは2つ使います)
　ハード制約 ： 必ず守りたい制約
　ソフト制約 ： できるだけ守りたい制約
✓レベル毎にスコアが計算される
✓スコアが高いほど最適解に近いと判断される
✓ハード制約のスコアがソフト制約のスコアよりも重要視される

◆シフト作成の設計の1例

ハード制約
- ✓ すべてのシフトに従業員が割当てられること
- ✓ 同一日に同じ従業員が複数のシフトに割当てられないこと

※満たさない場合はhard = -1

ソフト制約
- ✓ できるだけ各シフトが均等に割当てられること
- ✓ できるだけ夜シフトの翌日に朝シフトが割当てられないこと

※満たさない場合はsoft = -1

© Hitachi Solutions, Ltd. 2015. All rights reserved.

3-7. (4) スコア計算を実装する1

HITACHI
Inspire the Next

スコア計算はJavaでも実装できるが以下の点でDRLがお勧め

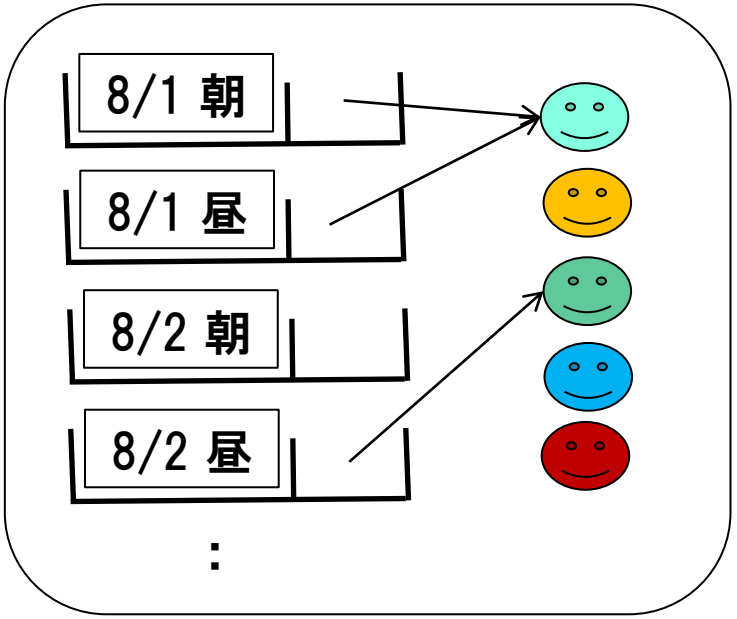
- ✓制約を外出しできるので、制約の追加・変更が容易になる
- ✓制約を直観的に実装できる

DRLにおけるスコア計算の記述形式

```
rule “ルール名“
    when [条件を記載]
    then  [アクション(スコア計算の処理)を記載]
end
```

ルールエンジン(Drools)と連携したスコア計算の処理

- ✓オブジェクトをワーキングメモリに入れる
- ✓OptaPlannerが組合せを変更する
- ✓ルールの条件に該当するすべての
  オブジェクトを抽出
- ✓抽出されたオブジェクトのアクション
  (スコア計算)を実行する



ワーキングメモリ

© Hitachi Solutions, Ltd. 2015. All rights reserved.

3-8. (4) スコア計算を実装する2

HITACHI
Inspire the Next

ScoreRules.drl

```
global HardSoftScoreHolder scoreHolder;
//すべてのシフトに従業員が割当てられること
//  デフォルトではOptaPlannerはすべての変数に値を割当てるため
//  このルールの実装は不要

//同一日に同じ従業員が複数のシフトに割当てられないこと
rule "oneShiftPerDay"
    when
        $left : ShiftAssignment(employee != null , $employee : employee, $shift : shift)
        $right : ShiftAssignment(employee == $employee, shift.day == $shift.day)
    then
        scoreHolder.addHardConstraintMatch(kcontext, -1);
End
// ソフト制約は省略
```

© Hitachi Solutions, Ltd. 2015. All rights reserved.

3-9. (5) 探索アルゴリズムを決める

HITACHI
Inspire the Next

モデルの特性を考慮して決める
分からなければワークベンチで評価する

ビルトインされている検索アルゴリズム

◆Exhaustive Search (ES)
- ➢Brute Force
- ➢Branch And Bound

◆Construction Heuristics (CH)
- ➢First Fit (Decreasing)
- ➢Weakest/Strongest Fit
- ➢Cheapest Insertion

◆Local Search (LS)
- ➢Hill Climbing
- ➢Tabu Search
- ➢Simulated Annealing
- ➢Late Acceptanc
- ➢Step Counting Hill Climbing

✓ESは総当たりで組合せを探索
✓効率が悪いので普通は使わない

✓CHは初期解を得るためのアルゴリズム

✓LSは組合せを少しずつ変えていき
より良い解を探す探索アルゴリズム
✓初期解が必要なのでCHと合わせて使用

© Hitachi Solutions, Ltd. 2015. All rights reserved.

3-10. (6) 終了条件を決める

HITACHI
Inspire the Next

終了条件の例

secondsSpentLimit	指定の時間が経過したら終了
unimprovedSecondsSpentLimit	スコアが改善せず指定の時間が経過したら終了
bestScoreLimit	あるスコア以上になったら終了
stepCountLimit	xxステップ実行したら終了
unimprovedStepCountLimit	スコアが改善せずxxステップ実行したら終了

✓上記の終了条件をANDまたはORで組合せることも可能
✓別スレッドから実行を終了させるためのスレッドセーフな
　メソッドも提供されている

© Hitachi Solutions, Ltd. 2015. All rights reserved.

3-11. (7) 設定ファイルを作成する

HITACHI
Inspire the Next

SolverConfig.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<solver>
  <solutionClass>package.ShiftSolution</solutionClass>
  <entityClass>package.ShiftAssignment</entityClass>
  <scoreDirectorFactory>
    <scoreDefinitionType>HARD_SOFT</scoreDefinitionType>
    <scoreDrl>package/ScoreRules.drl</scoreDrl>
  </scoreDirectorFactory>
  <constructionHeuristic>
    <constructionHeuristicType>FIRST_FIT</constructionHeuristicType>
  </constructionHeuristic>
  <localSearch>
    <acceptor><entityTabuSize>7</entityTabuSize></acceptor>
  </localSearch>
 <termination>
    <secondsSpentLimit>30</secondsSpentLimit>
  </termination>
</solver>
```

管理クラス、箱クラスを指定

制約を指定

CHのアルゴリズムを指定

LSのアルゴリズムを指定

終了条件を指定

© Hitachi Solutions, Ltd. 2015. All rights reserved.

3-12. (8) 実行する

HITACHI
Inspire the Next

ShiftCreatorMain.java

```
public static void main(String[] args) {
    SolverFactory solverFactory = SolverFactory.createFromXmlResource(
            “package/SolverConfig.xml”);
    Solver solver = solverFactory.buildSolver();

    //Employeeオブジェクト、ShiftAssignmentオブジェクトを生成して
    //管理クラス(ShiftSolution)にセットする
    ShiftSolution unsolved = createShiftSolution();

    solver.solve(unsolved);
    ShiftSolution solved = (ShiftSolution) solver.getBestSolution();

    //結果を表示する処理など
```

✓設定ファイルを読み込ませてSolverオブジェクトを生成する
✓solver.solve()で計算が開始される
✓最適解を取得するにはsolver.getBestSolution()を使用する
　スコアが最も良い組合せが格納された管理クラスが返る

© Hitachi Solutions, Ltd. 2015. All rights reserved.

3-13. (9) テスト・デバッグする

HITACHI
Inspire the Next

- ✓OptaPlannerはslf4jを使用しているので、お好みのログ実装・アダプタを
  クラスパスに追加してログ出力する
- ✓ログレベルをdebugにするとステップ毎の値の変更とスコアが出力される
- ✓以下の観点でテストする
  - •値の変更(move)が意図した通りに動いていること
  - •スコア計算が意図した通りに動いていること
- ✓ワークベンチを使って処理時間と得られる解の精度を分析する
- ✓必要に応じてチューニングする

```
INFO  Solving started: ...
DEBUG     CH step (0), ... score (-9hard/-1soft), ... picked move (0801_asa@null => [Employee-0]).
DEBUG     CH step (1), ... score (-8hard/-2soft), ... picked move (0801_hiru@null => [Employee-1]).
 :
INFO  Construction Heuristic phase (0) ended: ... best score (0hard/-10soft).
DEBUG     LS step (0), ... new best score (0hard/-9soft) ... picked move ([0801_hiru@Employee-1] => [Employee-4]).
DEBUG     LS step (1), ... new best score (0hard/-8soft) ... picked move ([0802_yoru@Employee-1] => [Employee-2]).
 :
INFO  Local Search phase (1) ended: step total (13), time spent (30000), best score (0hard/-1soft).
INFO  Solving ended: ... best score (0hard/-1soft) ...
```

© Hitachi Solutions, Ltd. 2015. All rights reserved.

参考サイト・書籍

HITACHI
Inspire the Next

■OptaPlanner コミュニティサイト
http://www.optaplanner.org/

■OptaPlannerによる組み合わせ最適化
http://www.ogis-ri.co.jp/otc/hiroba/technical/optaplanner/

■tokobayashiの日記
http://d.hatena.ne.jp/tokobayashi/searchdiary?word=%2A%5BOptaPlanner%5D

■Play Integration
http://playintegration.blogspot.jp/search?q=OptaPlanner

■簡単そうで難しい組合せ最適化
http://www-or.amp.i.kyoto-u.ac.jp/open-campus-04.pdf

■久保 幹雄, J.P.ペドロソ(2009)『メタヒューリスティクスの数理』共立出版.

■穴井 宏和(2013)『数理最適化の実践ガイド』講談社.

© Hitachi Solutions, Ltd. 2015. All rights reserved.

他社所有商標に関する表示

HITACHI
Inspire the Next

・Javaは、Oracle Corporation 及びその子会社、関連会社の米国及びその他の国における登録商標です。
・JBoss、Red Hatは、Red Hat, Inc. の米国およびその他の国における登録商標または商標です。
・Apacheは、Apache Software Foundationの登録商標または商標です。
・Linux は、Linus Torvalds 氏の日本およびその他の国における登録商標または商標です。
・Macは、米国Apple Inc.の米国およびその他の国における登録商標または商標です。

© Hitachi Solutions, Ltd. 2015. All rights reserved.

HITACHI
Inspire the Next

END

OSSプランニングエンジン
OptaPlannerを使ってみよう！

© Hitachi Solutions, Ltd. 2015. All rights reserved.