スケジューリング最適化コンポーネント

OptSeq

ユーザーズガイド

LOG OPT Co., Ltd.

ご注意

目 次

1 はじめに

OptSeq は，資源制約付きスケジューリング問題を最適化するための ActiveX コンポーネントです．OptSeq は，手法として禁断探索法ベースの発見的解法を用いています．指定された時間内で見つけることが出来た最良の解を出力します．OptSeq では，スケジュールの要素として，作業（activity）と資源（resource）を扱います．また，それぞれの作業はいくつかの処理方法（モード，mode）を選ぶことが出来ます．OptSeq では計画期間として 1 から始まる整数を扱います．なお，この説明書ではプログラム例を説明する際に Visual Basic コードを用いています．

本説明書の構成は以下の通りです．

2 節では個々のメソッドの詳細を説明します．

3 節では，スケジューリング問題を解く際に，それぞれのメソッドを使う順番を説明します．

2 メソッド

OptSeq はインターフェースとしてメソッドのみを用意しています．プロパティの設定はできません．メソッドは主に，動作設定メソッド，問題入力メソッド，求解メソッド，解出力メソッドに分類できます (表 1 参照)．

表 1: メソッドの分類

分類	メソッド
動作設定	setHorizon, setMaxCpuTime
問題入力	addRes, addResource, setResAmount, setResWeight, addMode, openModeRes, addResAmountOfMode, closeModeRes, addAct, addActivity, addMode2Act, setPrecedence, setPrecWithMeantime, setImmediatePrecedence, addSinkActivity, addCnstrnt, addTerm2Cnstrnt, setSignRightHand
求解	solveWithCnstrnt, solve
解出力	getActMode, getStartTime

以下に個々のメソッドの詳細を示します．

2.1 `short setHorizon(long horizon)`

引数: 計画期間

返値: 正常終了 (0)，計画期間が範囲外 (-10)

機能: 最大計画期間を設定します．設定できる計画期間の範囲は 1〜2147483647(INT_MAX) です．必ず設定してください．

例: `ret = 問題.setHorizon(365)` ’「問題」の計画期間は 365 日 (日単位の計画の場合)

2.2 short setMaxCpuTime(double cpu_time)

引数: 最大計算時間 (秒)

返値: 正常終了 (0)，最大計算時間が範囲外 (-10)

機能: 最大計算時間を秒単位で設定します．設定できる最大計算時間の範囲は 1〜3600 です．必ず設定してください．

例: `ret = 問題.setMaxCpuTime(60)` ’「問題」の解を 1 分以内に得たい

2.3 addRes(void)

引数: なし

返値: 資源 ID(> 0)

機能: 資源を追加します．

備考: このメソッドは資源を追加するだけです．資源量は全計画期間で 0 に設定されます．資源重みは 0 に設定されます．

例: 省略

2.4 addResource(long amount)

引数: 資源量

返値: 資源 ID(> 0)

機能: 使用量を指定した資源を追加します．

備考: 資源重みは 0 に設定されます．

例: 省略

2.5 setResAmount(short resID, long head, long tail, long amount)

引数: 資源 ID，設定期間の始まり，設定期間の終わり，設定量

返値: 正常終了 (0)，資源 ID が範囲外 (-11)，始まりが範囲外 (-12)，終わりが範囲外 (-13)，設定量が範囲外 (-14)

機能: 指定された期間の資源量を設定します．

備考: このメソッドは複数回使用できます．同じ期間に対して複数回使用した場合，最後の設定が有効になります．

例: 省略

2.6 setResWeight(short resID, long head, long tail, long weight)

引数: 資源ID，設定期間の始まり，設定期間の終わり，設定重み

返値: 正常終了(0)，資源IDが範囲外(-11)，始まりが範囲外(-12)，終わりが範囲外(-13)，設定重みが範囲外(-14)

機能: 指定された期間の資源の重み（重要度）を設定します．

備考: このメソッドは複数回使用できます．同じ期間に対して複数回使用した場合，最後の設定が有効になります．

例: 省略

2.7 addMode(long time, short res_num)

引数: 処理時間，使用する資源の種類数

返値: モードID(> 0)，処理時間が範囲外(-10)

機能: モードを追加します．

備考: このメソッドはモードを追加するだけです．処理時間は設定されますが，処理に必要な資源は設定されません．ただし，後に資源を設定するため，使用する資源の種類数を入力します．

例: 省略

2.8 openModeRes(short modeID, short resID)

引数: モードID，資源ID

返値: 正常終了(0)，モードIDが範囲外(-11)，資源IDが範囲外(-12)

機能: 処理に必要な資源のモードへの追加を開始します．

備考: このメソッドは資源IDを指定して資源の追加を開始するだけです．処理に必要な資源量は設定されません．

例: 省略

2.9 addResAmountOfMode(long head, long tail, long amount)

引数: 設定時刻の始まり，設定時刻の終わり，設定量

返値: 正常終了 (0)，モード ID が範囲外 (-11)，資源 ID が範囲外 (-12)，設定量が範囲外 (-13)，始まりが範囲外 (-14)，終わりが範囲外 (-15)

機能: 処理に必要な資源量を追加します.

備考: このメソッドは複数回使用できます．同じ期間に対して複数回追加した場合，使用資源量は加算されます.

例: 省略

2.10 closeModeRes()

引数: なし

返値: 正常終了 (0)

機能: 処理に必要な資源量のモードへの追加を終了します.

備考:

例: 省略

2.11 addAct(void)

引数: なし

返値: 作業 ID(> 0)

機能: 作業を追加します.

備考: このメソッドは作業を追加するだけです．作業のモードはまったく設定されません．作業には少なくとも 1 つはモードを追加してください.

例: 省略

2.12 addActivity (long proc_time)

引数: 作業の処理時間

返値: 作業 ID(> 0)

機能: 1 つの処理時間を指定したモードをもった作業を追加します.

例: 省略

2.13 addMode2Act(short actID, short modeID)

引数: 作業ID，モードID

返値: 正常終了(0)，作業IDが範囲外(-10)，モードIDが範囲外(-11)

機能: 作業にモードを追加します.

備考: 作業には少なくとも1つはモードを追加してください.

例: 省略

2.14 short setPrecedence(short act1, short act2)

引数: 先行する作業のID，後続する作業のID

返値: 正常終了(0)，対応する作業IDなし(-11)

機能: 先行順序関係を加えます．ここで指定された先行順序関係は絶対制約となります．先行する作業が終了しないと，後続する作業は開始できません.

例: 作業ID1 = 問題.addActivity(7)
作業ID2 = 問題.addActivity(5)
先行順序関係ID = 問題.setPrecedence(作業ID1, 作業ID2)

2.15 short setPrecWithMeantime(short act1, short act2, long time)

引数: 先行する作業のID，後続する作業のID，作業間の空き時間

返値: 正常終了(0)，対応する作業IDなし(-11)

機能: 先行順序関係を加えます．ここで指定された先行順序関係は絶対制約となります．先行する作業が終了してから，空き時間が経過した後でないと，後続する作業は開始できません.

例: 作業ID1 = 問題.addActivity(7)
作業ID2 = 問題.addActivity(5)
先行順序関係ID = 問題.setPrecWithMeantime(作業ID1, 作業ID2, 10)

2.16 short setImmediatePrecedence(short resID, short actID1, short actID2)

引数: 資源ID，先行する作業のID，後続する作業のID

返値: 正常終了(0)，対応する作業IDなし(-11)，対応する資源IDなし(-12)

機能: 直前先行順序関係を加えます．ここで指定された先行順序関係は絶対制約となります．直前先行制約は，第1引数で設定された共有する資源と同じ資源を使用する作業が，直前先行制約を付加された作業間に作業を開始することを禁止します．

例: 省略

2.17 short addSinkActivity()

引数: なし

返値: ダミー作業ID（> 0）

機能: ダミー作業を追加します．

備考: 先行順序関係の設定の後，目的関数の設定の前に呼び出してください．

例: 省略

2.18 short addCnstrnt(LPCTSTR name, long weight)

引数: 考慮制約式名，考慮制約式重み

返値: 考慮制約式ID（> 0），重みが範囲外（-10）

機能: 考慮制約式を追加します．

備考: なし

例: 省略

2.19 short addTerm2Cnstrnt(short kinkID, short actID, short modeID, long coeff)

引数: 項の種類(1:モードの選択に関する項，2:開始時間に関する項，3:終了時間に関する項，4:処理時間に関する項)，作業ID，モードID，項の係数

返値: 正常終了（0），異常終了（負）

機能: 考慮制約式に左辺項を追加します．追加できる項の種類は4種類です．

備考: モードの選択に関する項の場合のみ第3引数が考慮されます．

例: 省略

2.20 setSignRightHand(short signID, long rh)

引数: 等号・不等号種別（-1:≤, 0:=, 1:≥），右辺値

返値: 正常終了（0），異常終了（負）

機能: 考慮制約式の等号・不等号および右辺値を設定します.

備考: なし.

例: 省略

2.21 solveWithCnstrnt()

引数: なし

返値: 正常終了（0），異常終了（負）

機能: 禁断探索法を用いた発見的解法により得られる，最良の解を求めます.

備考: なし.

例: 省略

2.22 solve(long target)

引数: 最大完了時刻（メイクスパン）の目標値

返値: 正常終了（0），異常終了（負）

機能: 最大完了時刻を目標値からの遅れを最小にするような解を求めます.

備考: なし.

例: 省略

2.23 getActMode(short actID)

引数: 作業ID

返値: 指定された作業が採用したモードID（該当するモードがない場合は0）

機能: 作業が採用するモードを取得します.

備考: なし.

例: 省略

2.24 getStartTime(short actID)

引数: 作業ID

返値: 開始時間

機能: 指定したアクティビティの開始時間を取得します.

備考: なし.

例: 省略

3 メソッドを用いる順番

本コンポーネントにおいて，正しい解を得るためにはメソッドを正しい順番で使う必要があります．そうでない場合には異常終了してしまう場合がありますので，必ず正しい順番で使ってください.

以下にメソッドを用いる順番を述べます．大まかに分けると，動作設定，問題入力，求解，解出力の順番で使ってください.

まず最初に，動作設定のメソッド setHorizon, setMaxCpuTime を使ってください．使う順番は setHorizosn，setMaxCpuTime の順です．両方とも省略不可です.

次に問題入力のメソッド addRes, setResAmount, setResWeight, addMode, openModeRes, addResAmountOfMode, closeModeRes, addAct, addActivity, addMode2Act, setPrecedence, setImmediatePrecedence, addSinkActivity, addCnstrnt, addTerm2Cnstrnt, setSignRightHand を使ってください.

まず，資源を加えます．addRes は加える資源の数だけ繰り返し使ってください．1つの addRes に対して，setResAmount, setResWeight は複数使えます．設定したい資源量，資源重みを表現できるまで繰り返し使ってください.

次に，モードを加えます．addMode は加える資源の数だけ繰り返し使ってください．1つのモードに対して，複数の使用資源を設定できます．使用資源の数だけ openModeRes, closeModeRes を繰り返し使ってください．使用資源量は処理時間内で任意に設定できます．openModeRes, closeModeRes の1つの対に関して，設定したい資源量を表現できるまで addResAmountOfMode を繰り返し使ってください.

次に，作業を加えます．addAct は加える作業の数だけ繰り返し使ってください．1つの作業に対して複数のモードを設定できます．addMode2Act を使ってモードを作業に加えてください．次に先行順序制約を加えます．一般先行順序制約を加える場合には setPrecedence を，直前先行順序制約を加える場合には setImmediatePrecedence を使ってください．先行順序制約の数だけ繰り返し使ってください.

次に，ダミーの作業を加えます．addSinkActivity を使ってください．addSinkActivity は省略不可です.

次に考慮制約式を加えます．制約式の数だけ addCnstrnt を繰り返し使ってください．1つの制約式は複数の左辺項と等式・不等式と右辺項で構成されます．まず，addTerm2Cnstrnt により必要なだけ左辺項を

加えてください．次に，setSignRightHand により等式・不等式と右辺項を設定してください．このようにして各考慮制約式を構成します．考慮制約式を入力し終わったら，問題入力は完了です．

次に求解のメソッド solveWithCnstrnt を使ってください．

最後に解出力のメソッド getActMode, getStartTime を使ってください．getActMode は各作業どのモードを使うのか出力します．getStartTime は各作業の開始時間を出力します．