修 士 論 文

資源割り当て駆動パイプラインスケジューリングとその高位合成への応用

指導教官　金子峰雄 助教授

北陸先端科学技術大学院大学
情報科学研究科情報システム学専攻

萬屋 俊之

2001 年 2 月 15 日

目 次

図 目 次

表 目 次

第 1 章

はじめに

1.1 研究の背景と目的

集積回路システムの自動設計においては，設計階層の最も上位に位置するシステムレベルの記述言語からトップダウン的に，レジスタトランスファー（RT）レベル，論理レベルの順に下位に位置するそれぞれの階層の記述言語に自動変換する技術が用いられる．その中で高位合成はアルゴリズムレベル動作記述から，レジスタ，演算器，バスなどで構成された，レジスタ・トランスファー（RT）レベル動作記述を生成する設計技術である [1].

従来の高位合成システムにおいては，演算器の性能，面積が計算アルゴリズムの実行時間，回路面積を決める支配要因であるような VLSI(Very Large Scale Integrated circuit) を対象とし，コントロールステップ数，演算器数の最小化が主な最適化目標であった．そのため，資源制約型スケジューリングや時間制約型スケジューリングにより，コントロールステップ数や資源数の最小化を目的としたスケジューリングが行われ，その結果を受けて，演算の演算器への割り当て，データのレジスタへの割り当て等の資源割り当て，接続関係生成，レイアウト生成の順に実行される逐次最適化が多く採用されていた.

しかし，VLSI 製造技術の発展による回路素子の微細化に伴い，配線幅が 0.25 ミクロン以下の VLSI が製造されるようになってきており，この微細化の傾向は今後さらに進むことが予想されている．こうした deep submicron VLSI では，配線幅や配線長に依存する素子間の信号伝達遅延が，回路動作時間に占める割合が大きくなり，システム性能への影響も無視できなくなってきている．そのため配線による信号伝達遅延を考慮した設計を行うための，コンポーネント間接続関係あるいはフロアプランの最適性を設計の初期段階から考慮した高位合成が必要とされており，このような高位合成として設計の手戻り，あるいは資源割り当てを中心とする最適化などが検討されつつある [2][3].

このような設計アプローチにおいて，資源割り当てが指定された状況でのスケジューリング問題にしばしば遭遇するが [4][5][6]，その解法について十分に検討されているとは言えない．

本研究では高位合成の動作記述として繰り返しループを含むアルゴリズムのパイプラインスケジューリングにおいて，指定される資源割り当て情報（演算，データの演算器，レジスタへの割り当て）から要請される制約を制約枝（disjunctive arc）の形で先行制約グラフに反映させてパラメトリックスケジューリンググラフを構成するアプローチ及び，それに基づく効率的な厳密解法の一例および heuristics を用いた効率的解法の一例を示す．また，提案手法の 5 次楕円フィルタに対する資源割り当て空間探索をコアとするデータパス合成への適用例を示し，手法の有効性を確認する．

1.2 本論文の構成

先ず，第 2 章に VLSI 設計の高位合成について述べ，第 3 章で割り当て駆動スケジューリング問題について，第 4 章でスケジューリング問題の準備について，第 5 章で指定された資源割り当て情報から得られる制約枝を付加したスケジューリング手法について，第 6 章で，厳密解の探索および heuristics を用いた問題の解法について，第 7 章で，本手法を高位合成に適用したシステムの実装について，第 8 章にてまとめと今後の課題を述べる．

第 2 章

高位合成

2.1 高位合成とは

高位合成（データパス合成）は VLSI 記述階層の上位に位置し，要求される仕様の動作記述もしくはアルゴリズム記述を入力とし，演算器，レジスタ，バス等で構成されたレジスタトランスファレベル（RT レベル）のアーキテクチャ（データパス部）と制御回路・コード（コントロール部）を出力とする合成システムである．

入力であるアルゴリズム記述から必要な演算やデータを算出し，先行制約グラフを作成することが可能であり，この先行制約グラフを本研究の入力として用いることとする．先行制約グラフの頂点は演算及びデータを示し，枝はデータの流れを表すものとする．

高位合成は演算器・レジスタ指定問題，演算スケジューリング問題，演算器・レジスタ割り当て問題の 3 つの主要な部分問題を含んでいる．各部分問題の出力をもとに，RT レベル動作記述を生成する．なお，各部分問題は $\mathcal{NP}$ 完全なクラスに属し，入力サイズの増大に伴い最適解の探索は困難になることが知られている．図 2.1に従来型高位合成の構成図を示す．

主な部分問題の概略を以下に示す．

2.1.1 演算スケジューリング

演算スケジューリングは，各演算間のデータの依存関係を保ちつつ各演算の実行時刻を決定する処理である．すなわち，演算及びデータの時刻への割り当てである．時刻は 1 クロック時間のステップ単位で扱うことが多く，コントロールステップと呼ばれる．

各演算のスケジューリングは，考慮すべき制約により大きく次の 2 つに分類される．

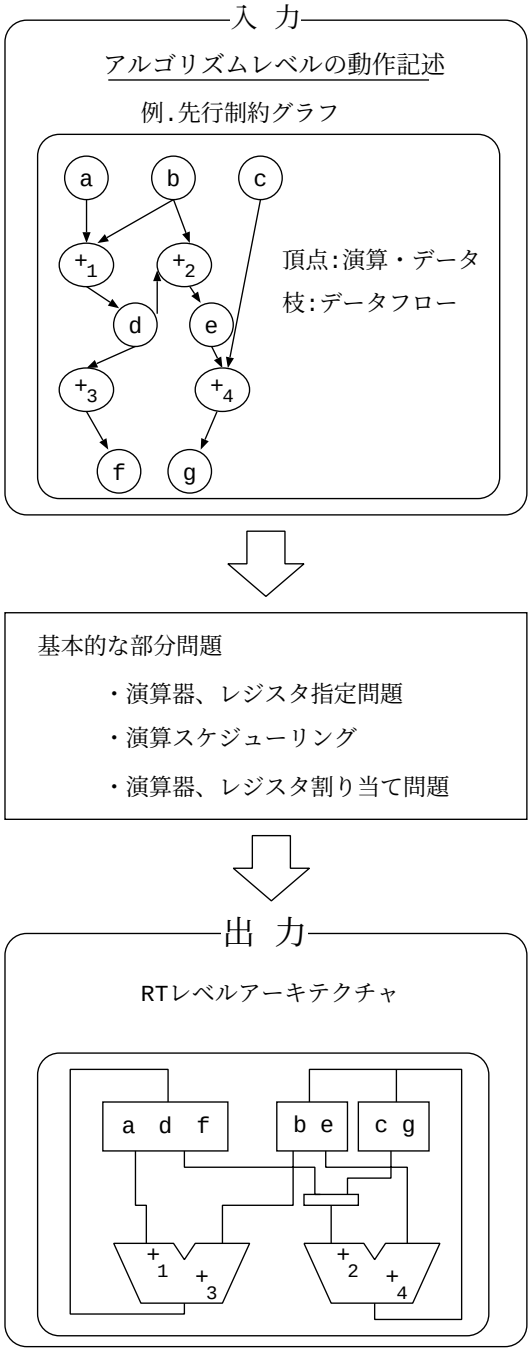

入 力
アルゴリズムレベルの動作記述
例.先行制約グラフ
a
b
c
+1
+2
d
e
+3
+4
f
g
頂点:演算・データ
枝:データフロー
基本的な部分問題
・演算器、レジスタ指定問題
・演算スケジューリング
・演算器、レジスタ割り当て問題
出 力
RTレベルアーキテクチャ
a d f
b e
c g
+1
+3
+2
+4

図 2.1: 高位合成の流れ

1. 資源制約型スケジューリング

   演算器の種類と数が与えられたときに，その制約内で処理時間を最小とするスケジューリング

2. 時間制約型スケジューリング

   全ての演算が実行終了するまでの期限時刻が与えられた時に，その制約内での演算器数を最小とするスケジューリング

これらのスケジューリングに対し，資源制約を定めずに各演算をできるだけ早く開始する ASAP（As Soon As Possible）スケジューリング，同様に資源制約を定めずにできるだけ遅く各演算を終了するようにスケジューリングを行う ALAP(As Late As Possible）スケジューリング等の情報を有効に活用できる．

2.1.2 資源割り当て

資源割り当ては，演算頂点を実際に演算を行う演算器に割り当て，データ頂点を実際にデータを格納するレジスタに割り当てる処理である．演算のタイプ（加算，乗算）に対し使用可能な演算器のタイプ（加算器，乗算器，ALU）が指定されており，それに使用可能な演算器を割り当てる必要がある．データも同様に様々なビット長のデータがあり，データのビット長より大きいビット長を格納することが可能なレジスタを指定する必要があり，それに使用可能なレジスタを割り当てる必要がある．ここで，演算に対し，演算器タイプとその数を指定する問題と，データに対し，レジスタタイプとその数を指定する問題は，演算器・レジスタ指定問題と呼ばれ，高位合成の部分問題として重要である．

各演算器，レジスタは同時に一つの演算，データしか扱えないことから，演算が実行されている時間区域が重なっている場合やデータが存在している時間区域が重なっている場合にはその演算やデータは同じユニットを使用することができないことになる．この演算が実行されている時間，データが存在している時間をそれぞれ演算の生存期間，データの生存期間と呼ぶ．これは演算のスケジューリングが決まった時点で演算の行われている時間が決定されるため，データが生成される時刻及びそのデータを使用する演算の実行時刻が決まればデータの生存期間も決定される．また生存期間が重ならない演算またはデータは同一の演算器またはレジスタを使用することができる．

2.2 従来手法の問題点

従来の高位合成手法では，先ず，与えられた先行制約グラフから，資源制約型スケジューリングもしくは時間制約型スケジューリングを行い，演算やデータの順序関係を生成し，その結果を受けて．演算の演算器，データのレジスタ割り当てを行い，結線関係生成，レイアウトと続く逐次処理が行われてきた．資源制約型スケジューリングにより，演算器などの資源数，時間制約型スケジューリングにより，演算器などの資源数は最適化される．

一方，近年の VLSI 微細加工技術の進歩に伴い，結線関係に起因する動作速度や消費電力などのシステム性能への影響が無視できなくなってきている．しかし，従来のスケジューリングを起点とした高位合成の逐次処理では，設計の初期段階において，結線関係を考慮することは困難であり，結線関係やそれに起因する信号遅延，消費電力，チップ面積等については，最適性の保証がなされないのが現状である（図 2.2）．

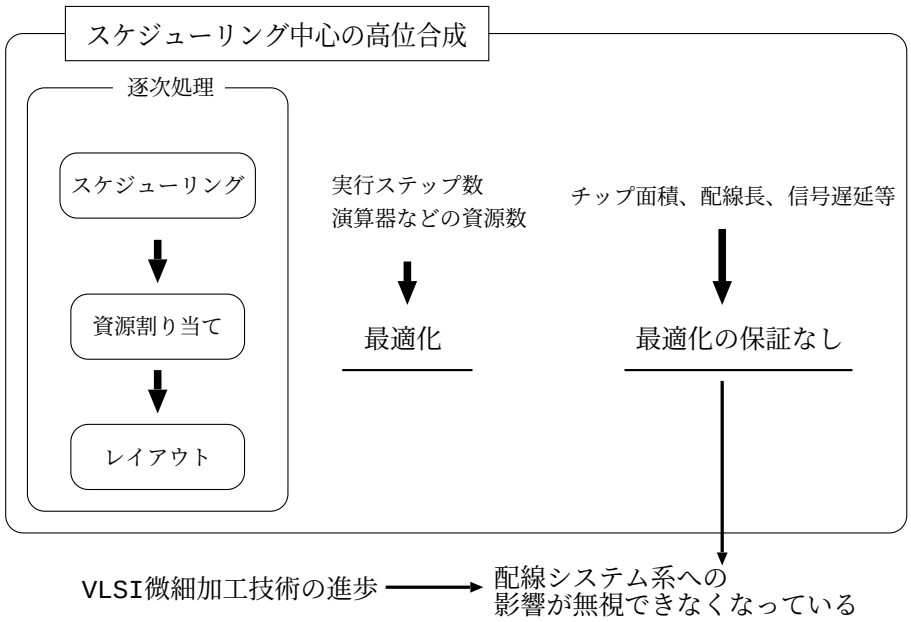


図 2.2: 従来手法による高位合成

2.3　資源割り当て空間探索をコアとするデータパス合成

VLSI 微細加工技術の進歩に伴って，配線の VLSI 性能への寄与の度合いが高くなって来ており，モジュール間接続関係やフロアプランを考慮したデータパス合成が必要となって来ている．これに対応するためには，高位合成の初期段階からモジュール間接続関係に配慮した最適化が必要と考えられる．こうしたことから，資源割り当て空間探索をコアとする合成手法が検討されている．

最初に，演算を演算器，データをレジスタに割り当てる資源割り当てを行い，その資源割り当て情報から要請される制約を満たしながらスケジューリングを行う．その結果からモジュール間接続関係の評価を行う．この操作を繰り返す．

概略を図 2.3に示す．

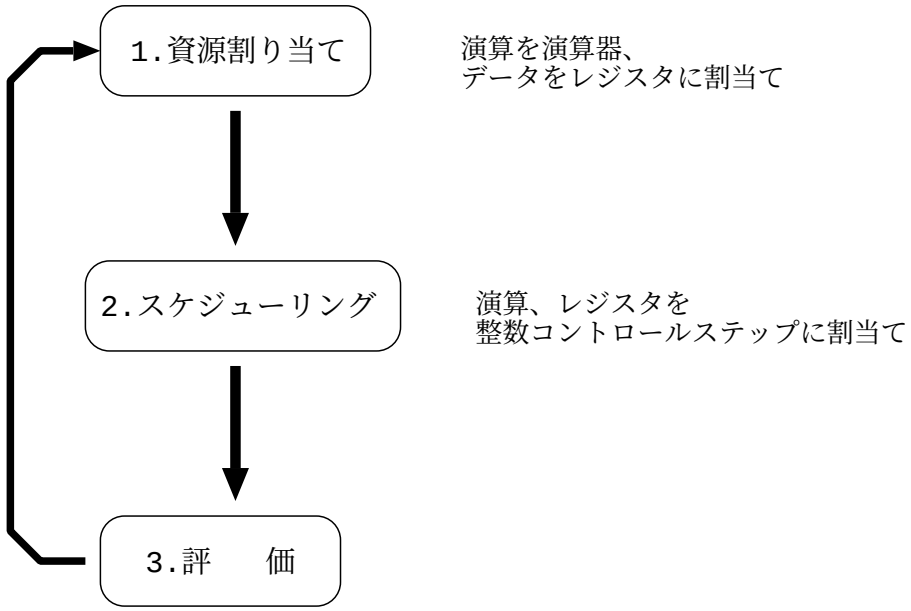


図 2.3: 提案手法による高位合成

第 3 章

割当て駆動スケジューリング問題

合成対象であるアルゴリズム記述中には，再帰的な計算過程を含むものが多い．こうした再帰的な計算アルゴリズムは通常，枝に遅延情報を持たせたサイクリックグラフとして表現される．グラフの頂点が演算を表すものとし，枝が先行関係を表すものとして，図3.1に示すグラフの中で，演算頂点 v は 1 回前の実行で生成された演算頂点 x の結果と，2 回前の実行で生成された演算頂点 y の結果を使用して演算を行うことを意味している．再帰的な計算アルゴリズムの実行においては，パイプライン化を行って，1 繰り返しブロックの計算時間が長くても，パイプラインの繰り返し周期を小さくすることで，全体としての計算時間を短縮できる．また，信号処理システムにおいては，連続して入力される信号を処理して，連続して信号を出すことが要求され，この入力あるいは出力信号の時間間隔は，サンプリング周期に対応する．リアルタイム処理ではアプリケーションが要求するサンプリング周期で次々に信号の入出力ができなければならない．リアルタイム信号処理システムでのサンプリング周期に対応する，パイプラインスケジューリングにおける繰り返し周期の短縮は非常に重要な問題であると考えられる．

再帰的な計算アルゴリズムをグラフの枝の遅延情報を使用せずに示すと，図 3.2のグラフで表される．このように遅延情報を使用せずに示した再帰的な計算を表すグラフは展開表示されたグラフと呼ばれる．展開表示されたグラフは有向サイクルが存在しない有向グラフ，すなわち DAG(Directed Acyclic Graph) となっている．

しかし，繰り返しの回数が動作時に定まる場合や，信号処理のような無限の繰り返しを想定するアルゴリズムでは展開表示することができない．また，信号処理などを実際に使用できる範囲で展開したとしてもグラフが巨大なものとなり，スケジューリングが困難になると共に，システム合成後の制御部のサイズも巨大なものとなり，実用的ではない．こうしたことから，展開表示を用いずに枝に遅延情報を持たせた，サイクリックな先行制約

グラフをそのまま用いたスケジューリングを行うのが一般的である．以上をふまえて，本研究では，資源割り当て空間探索をコアとするデータパス合成を想定し，そこでの資源割り当てを指定した状況での繰り返しループを含むアルゴリズムのパイプラインスケジューリング問題についての検討を行う．

具体的な問題の記述を以下に示す．

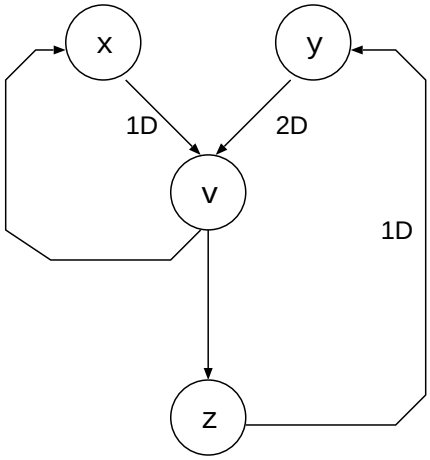


図 3.1: 枝に遅延情報を持つグラフ

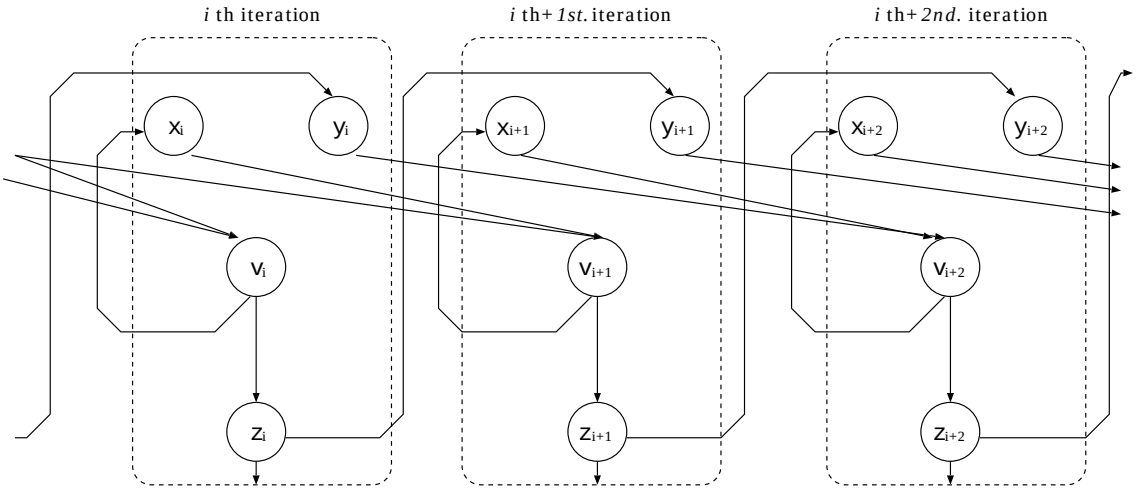


図 3.2: 遅延を持つグラフの展開表示

問題の記述

入力;

- 先行制約グラフ $G = (V_G, A_G)$;

  V_G は演算集合 V_O とデータ集合 V_D の和集合.

  A_G は演算頂点からそれが生成するデータ頂点へ向かう枝の集合 A_O とデータ頂点から演算頂点へ向かう枝の集合 A_I の和集合.

  なお，A_I は遅延関数（$D : A_I \to Z$）を持ち，$(d_i, o_j) \in A_I$ に対して，o_j の計算が $D(d_i, o_j)$ 回前の繰り返し実行中に生成されたデータ d_i を用いることを表す.

- 資源割当て $\rho : V_O \to \mathcal{F},\ \xi : V_D \to \mathcal{R}$;

  $\mathcal{F}$ は演算器の集合.

  $\mathcal{R}$ はレジスタの集合を表す.

- 演算実行時間 $e : V_O \to Z+$;

出力;

- $\sigma : V_O \to Z$

  σ は各演算の同一繰り返し実行中の実行開始コントロールステップを表すものとし，各演算は，繰り返し周期（Tr）と呼ばれる実行間隔で繰り返し実行される．すなわち，$o_i \in V_O$ は $\sigma(o_i) + kTr,\ k = \ldots, 0, 1, 2, \ldots$ にて実行される.

  但し，

  1. G にて指定される演算，データの先行制約を満足する.
  2. 同一演算器に割当てられる演算同士，あるいは同一レジスタに割当てられるデータ同士の生存期間は重複しない．但し，データはそれを生成する演算の終了コントロールステップの次のコントロールステップから，それを使う演算の終了コントロールステップまでの間，レジスタに保持されるものとする.

なお，本研究では，ブロックスケジューリング，チェイニングは考慮しない.

先行制約を満たすスケジューリングは以下の制約式を同時に満足する σ を求める問題としてとらえることができる.

$$\forall (d_i, o_j) \in A_I,$$
$$\sigma(o_j) \quad \geq \quad \sigma(p(d_i)) + e(p(d_i)) - D(d_i, o_j)Tr$$

但し，$p(d_i)$ は，d_i を生成する演算を表す．すなわち，

$$(p(d_i), d_j) \in A_O \text{ かつ } o \neq p(d_i) \Rightarrow (o, d_i) \notin A_O$$

以上の先行制約関係を図示すると，図（3.3）に表される．

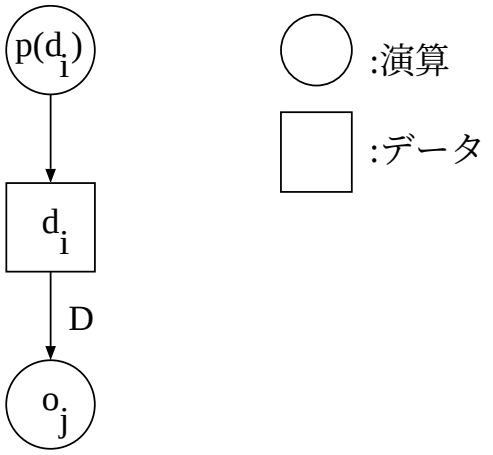


図 3.3: 先行制約関係

またリタイミングの自由度 $\phi : V_O \to Z$ を考慮し，以下の制約式を同時に満足する (ϕ, σ) を求める問題とすることも多い．

$$\begin{aligned} & \forall (d_i, o_j) \in A_I, \\ \sigma(o_j) + \phi(o_j)Tr \quad &\geq \quad \sigma(p(d_i)) + \phi(p(d_i))Tr \\ & \quad +e(p(d_i)) - D(d_i, o_j)Tr \end{aligned}$$

また資源割当てが与えられていない場合，パイプラインスケジューリングに関して以下の事項が知られている．

補題 1：全ての有向閉路中の遅延の総和が 1 以上ならば，スケジューリング可能な Tr が存在する

補題 2： 繰り返し周期 Tr に関し，以下の不等式が成り立つ．

$$Tr \geq \underset{\text{cycle } C}{\text{MAX}} \left\lceil \frac{\sum_{o_m \in V(C)} e(o_m)}{\sum_{(d_m, o_n) \in A(C)} D(d_m, o_n)} \right\rceil$$

補題 3：$\forall o \in V_O,\ e(o) \geq Tr$ であって，資源数が十分に多い場合，繰り返し周期をその下界値とするスケジューリングが存在する

第 4 章

スケジューリングのための準備

4.1 スケジューリンググラフ

スケジューリング問題を取り扱うに当たり，各演算の実行開始，終了コントロールステップを明示的に表現するために，スケジューリンググラフ $G_S = (V_S, A_S)$ を導入する．

まず，表記を判り易くするために，$\lfloor o_i \rfloor$ と $\lceil o_i \rceil$ をそれぞれ，$o_i \in V_O$ の演算開始と演算終了に対応する頂点とし，

$$V_{\lfloor\ \rfloor} = \{\lfloor o_i \rfloor \mid o_i \in V_O\}$$

$$V_{\lceil\ \rceil} = \{\lceil o_i \rceil \mid o_i \in V_O\}$$

とする．

スケジューリンググラフ $G_S = (V_S, A_S)$ は頂点集合を $V_S = V_{\lfloor\ \rfloor} \cup V_{\lceil\ \rceil}$ とし，A_S を先行制約枝とするグラフである．特に A_S は定数制約枝集合（A_C）と下限制約枝集合（$A_{\leq}$）とから成る．

また 2 つの枝重み関数

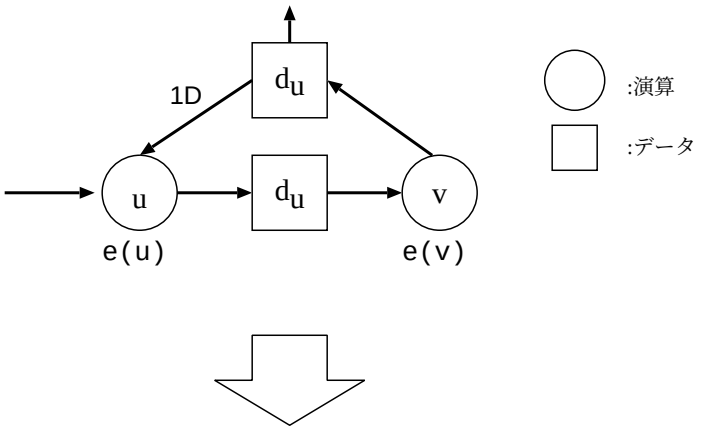
$$W \ : \ A_C \cup A_{\leq} \to N$$

$$D_S \ : \ A_{\leq} \to Z$$

が指定される．

図 4.1に，先行制約グラフからスケジューリンググラフの生成の摸式図を示す．

先行制約グラフ



スケジューリンググラフ

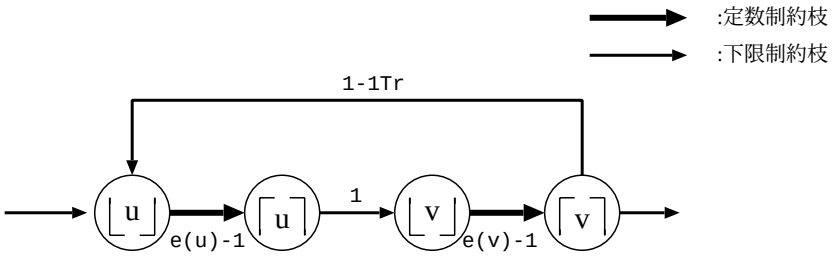


図 4.1: 先行制約グラフからスケジューリンググラフの生成

ここで，G_S 上でのスケジューリング $\sigma : V_S \to Z$ を考える時，定数制約枝 $(p, q) \in A_C$ について，

$$\sigma(q) = \sigma(p) + W(p, q)$$

また下限制約枝 $(r, s) \in A_{\leq}$ について．

$$\sigma(s) \geq \sigma(r) + W(r, s) - D_S(r, s)Tr$$

が要請されるものとする．

特に与えられた先行制約グラフ $G = (V_G, A_G)$ に対して，

$$A_C = \{(\lfloor o_i \rfloor, \lceil o_i \rceil) \mid o_i \in V_O\}$$

$$W(\lfloor o_i \rfloor, \lceil o_i \rceil) = e(o_i) - 1$$

$$A_{\leq} = \{(\lceil o_i \rceil, \lfloor o_i \rfloor) \mid \exists d \ s.t. \ o_i = p(d), \ (d, o_j) \in A_I\}$$

$$W(\lceil o_i \rceil, \lfloor o_i \rfloor) = 1$$

$$D_S(\lceil p(d) \rceil, \lfloor o_j \rfloor) = D(d, o_j)$$

にて与えられるグラフを初期スケジューリンググラフ G_{S0} と呼ぶ

4.2 ASAP スケジューリングと ALAP スケジューリング

G_S 上でのスケジューリングにおいて，特にある頂点 $v \in V_S$ を基準点とし，$\sigma(v) = 0$ と固定したスケジューリングを $\sigma_v : V_S \to Z$ とする．またこの時，基準点を v とする ASAP スケジューリング $\sigma_{ASAP\ v}$，ALAP スケジューリング $\sigma_{ALAP\ v}$ を次のように定義する．

$$\sigma_{ASAP\ v}(p) ='' G_S\text{上の } v \text{ から } p \text{ への最大パス長}''$$

$$\sigma_{ALAP\ v}(p) = -''G_S\text{上の } p \text{ から } v \text{ への最大パス長}''$$

但し，G_S 上の枝の長さを $(p, q) \in A_C$ に対して，$W(p, q)$，$(r, s) \in A_{\leq}$ に対して，$W(r, s) - D_S(r, s)Tr$ と定義する．

従って，最大パス長は

$$\sum W(p, q) + \sum W(r, s) - \sum D_S(r, s)Tr$$

で表される．

この時以下の補題が成り立つ．

補題 4：

$$\sigma_{ASAP\ v}(p) \leq \sigma_v(p) \leq \sigma_{ALAP\ v}(p)$$

4.3 最大パス長出力アルゴリズム

本研究における最大パス長を求める問題は，スケジューリンググラフの枝重みの正負を逆転させると，最小パス長を求める問題に変換される．また，一つの演算器に全ての演算を割り当てる場合もしくは，一つのレジスタに全てのデータを割り当てる場合，全点間に対して，ASAP スケジューリング，すなわち最小パスを求める必要がある．そこで，枝重みが負である有向グラフに対して適用でき，かつ全点間の最小パスを求める計算量が最小となる Floyd のアルゴリズムを採用した．

4.3.1 Floyd のアルゴリズム

以下にアルゴリズムを示す．

有向グラフ $G = (V, E)$ において，$V = \{v_1, v_2, \ldots, v_n\}$ のように，各頂点には 1 から n までの番号がつけられているものとする．

```
   procedure FLOYD:
   begin
1  for all i and j (1 ≤ i, j ≤ n and i ≠ j) do
2      if (v_i, v_j) ∈ E then
3          W_{i,j} ← w(v_i, v_j)
       else
4          W_{i,j} ← ∞;
5  for all i (1 ≤ i ≤ n) do
6      W_{ii} ← 0;
7  for all i and j (1 ≤ i, j ≤ n) do
8      father (i, j) ← i;
9  for k ← 1 until n do
       begin
10         for all i and j (1 ≤ i, j ≤ n) do
11           if W_{ij} > W_{ik} + W_{kj};
                 begin
12                   W_{ij} ← W_{ik} + W_{kj};
13                   father (i, j) ← father (k, j)
                 end
14         if there is any W_{ii} < 0 then no solution is possible
       end
   end
```

4.3.2 Floyd のアルゴリズムの計算量

示したアルゴリズム中の 9〜14 行目のループが各 k ($1 \leq k \leq n$) に対して，1 度ずつ実行され，その各段階で 10〜13 行目のループが全ての i, j $(1 \leq i, j \leq n)$ に対して 1 回ずつ実行される．従って，この手続き全体の計算量は

$$\mathcal{O}(n^3)$$

である．

第 5 章

制約枝追加とスケジューリング手法

5.1 資源割り当てからの制約

G 中の 2 つの演算 o_a と o_b が同一の演算器に割り当てられた場合，それら演算の生存期間の重なりを回避するために次の制約条件が必要かつ十分となる．すなわち，ある整数 K_{ab} が存在し，

1. i 回目の繰り返し実行における o_a の生存期間が $i - K_{ab}$ 回目の繰り返し実行における o_b の生存期間に先行する（図 5.1-(1)）．かつ,
2. $i - K_{ab}$ 回目の繰り返し実行における o_b の生存期間が，$i + 1$ 回目の繰り返し実行における o_a の生存期間に先行する（図 5.1-(2)）

今 K_{ab} を未知整数変数として，上記 1,2 より，直ちに以下を得る．

$$\begin{aligned} \sigma(\lfloor o_b \rfloor) + (i - K_{ab})Tr &> \sigma(\lceil o_a \rceil) + iTr \\ \Rightarrow \sigma(\lfloor o_b \rfloor) &> \sigma(\lceil o_a \rceil) + K_{ab}Tr \end{aligned} \tag{5.1}$$

かつ,

$$\begin{aligned} \sigma(\lfloor o_a \rfloor) + (i + 1)Tr &> \sigma(\lceil o_b \rceil) + (i - K_{ab})Tr \\ \Rightarrow \sigma(\lfloor o_a \rfloor) &> \sigma(\lceil o_b \rceil) - (1 + K_{ab})Tr \end{aligned} \tag{5.2}$$

これら 2 つの制約は，スケジューリンググラフ G_S に重みを変数 K_{ab} についての関数とする下限制約枝 $(\lceil o_a \rceil, \lfloor o_b \rfloor), (\lceil o_b \rceil, \lfloor o_a \rfloor)$

但し，

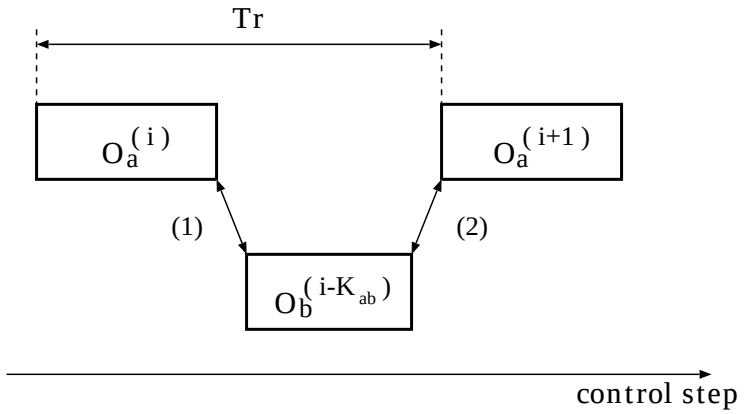


図 5.1: 同一演算器に割り当てられた 2 つの演算の生存期間の重なり回避のための制約

$$D_S(\lceil o_a \rceil,\ \lfloor o_b \rfloor) = -K_{ab}$$

$$D_S(\lceil o_b \rceil,\ \lfloor o_a \rfloor) = 1 + K_{ab}$$

$$W(\lceil o_a \rceil,\ \lfloor o_b \rfloor) = W(\lceil o_b \rceil,\ \lfloor o_a \rfloor) = 1$$

を付加することで表現できる（図 5.2）.

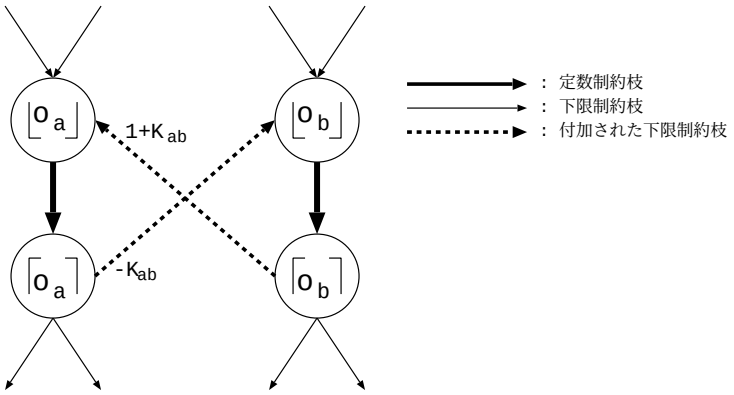
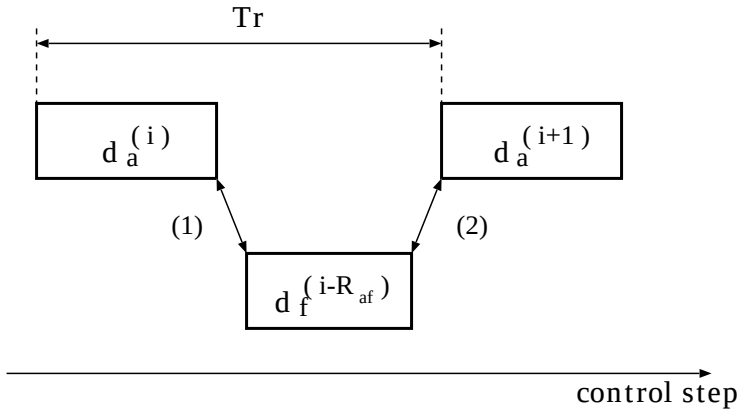


図 5.2: 同一演算器に割り当てられた 2 つの演算間の制約枝

一方，2 つの演算 o_a と o_f が生成する 2 つのデータ d_a と d_f が同一のレジスタに割り当てられた場合，それらのデータの生存期間の重なりを回避するために，次の制約条件が必要（かつ十分）となる．すなわち，ある整数 R_{af} が存在し，

1. i 回目の繰り返し実行中に生成された d_a の生存期間が $i - R_{af}$ 回目の繰り返し実行中に生成された d_f の生存期間に先行する（図 5.3-(1)）．かつ，

2. $i - R_{af}$ 回目の繰り返し実行中に生成された d_f の生存期間が，$i + 1$ 回目の繰り返し実行中に生成された o_a の生存期間に先行する（図 5.3-(2)）



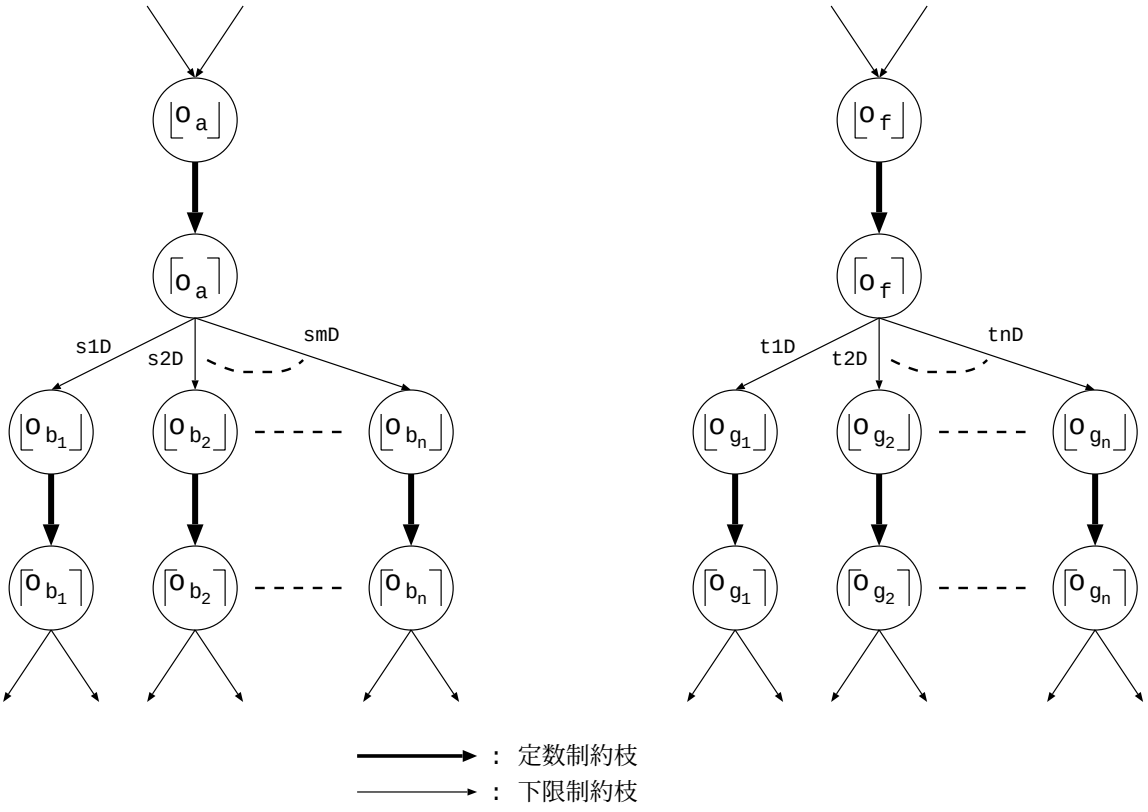


図 5.4: 同一レジスタに割当てる 2 データに関するスケジューリンググラフ

これより，上記 1,2 のデータの生存期間が重ならない条件は R_{af} を未知変数として，以下の通り記述できる．

$$\begin{aligned} \underset{x}{\mathrm{MAX}}\,[\sigma(\lceil o_{b_x}\rceil)+(i+s_x)Tr] &< \sigma(\lceil o_f\rceil)+(i-R_{af})Tr+1 \\ \Rightarrow \sigma(\lceil o_f\rceil) &\geq \underset{x}{\mathrm{MAX}}\,[\sigma(\lceil o_{b_x}\rceil)+(R_{af}+s_x)Tr] \end{aligned} \tag{5.3}$$

$$\begin{aligned} \underset{y}{\mathrm{MAX}}\,[\sigma(\lceil o_{g_y}\rceil)+(i-R_{af}+t_y)Tr] &< \sigma(\lceil o_a\rceil)+(i+1)Tr+1 \\ \Rightarrow \sigma(\lceil o_a\rceil) &\geq \underset{y}{\mathrm{MAX}}\,[\sigma(\lceil o_{g_y}\rceil)+(t_y-R_{af}-1)Tr] \end{aligned} \tag{5.4}$$

以上の制約は，演算器割り当ての場合と同様に，スケジューリンググラフ G_S に下限制約枝 $(\lceil o_{b_x}\rceil,\ \lfloor o_f\rfloor), x=1,2,\ldots,m$，$(\lceil o_{g_y}\rceil,\ \lfloor o_a\rfloor), y=1,2,\ldots,n$，

但し，

$$D_S(\lceil o_{b_x}\rceil,\ \lceil o_f\rceil) = -D_S(\lceil o_a\rceil,\ \lfloor o_{b_x}\rfloor) - R_{af}$$

$$D_S(\lceil o_{g_y} \rceil,\ \lceil o_a \rceil) = -D_S(\lceil o_f \rceil,\ \lfloor o_{g_y} \rfloor) + 1 + R_{af}$$

$$W(\lceil o_{b_x} \rceil,\ \lceil o_f \rceil) = W(\lceil o_{g_y} \rceil,\ \lceil o_a \rceil) = 0$$

を付加することで表現できる（図 5.5）

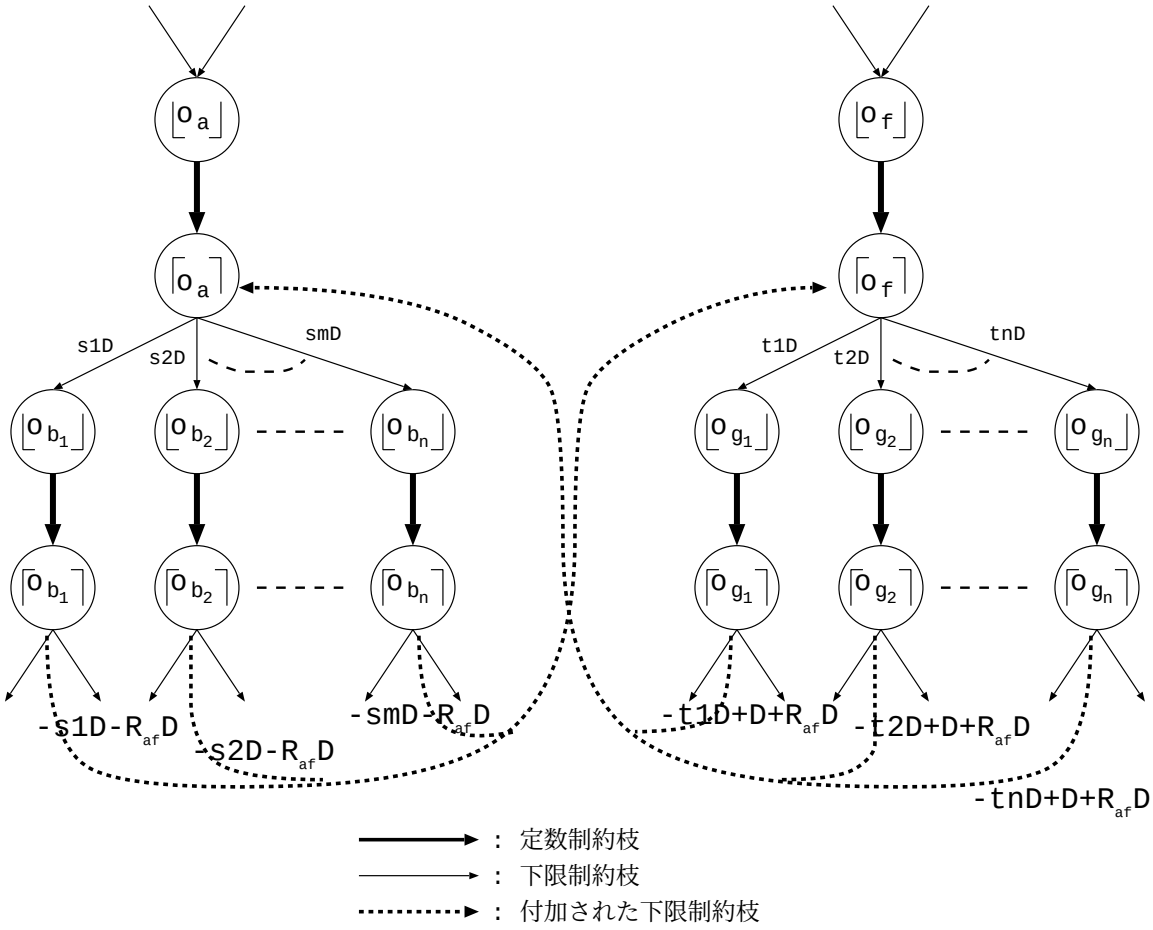


図 5.5: 同一レジスタに割当てられた 2 データに関する制約枝

一方．o_c にて生成されるデータ d_c がレジスタを単独で使用する場合には連続して生成される d_c の生存期間に重なりがあってはならない．

このことから，d_c を使用する演算を $o_{d_z},\ z = 1, 2, \ldots, n$，とし，$d(\lceil o_c \rceil, \lfloor o_{d_z} \rfloor) = y_z$ として，

$$\underset{z}{\mathrm{MAX}}\ [\sigma(\lceil o_{d_z} \rceil) + (i + y_z)Tr] < \sigma(\lceil o_c \rceil) + (i + 1)Tr$$

$$\Rightarrow \sigma(\lceil o_c \rceil) \geq \underset{z}{\mathrm{MAX}}\ [\sigma(\lceil o_{d_z} \rceil) + (y_z - 1)Tr]$$

が要請される．これについてもスケジューリンググラフに下限制約枝 $(\lceil o_{d_z} \rceil,\ \lceil o_c \rceil), z = 1, 2, \ldots, n$,

　但し,

$$D_S(\lceil o_{d_z} \rceil,\ \lceil o_c \rceil) = -D_S(\lceil o_c \rceil,\ \lfloor o_{d_z} \rfloor) + 1$$

$$W(\lceil o_{d_z} \rceil,\ \lceil o_c \rceil) = 0$$

を付加することで表現される（図 5.6）.

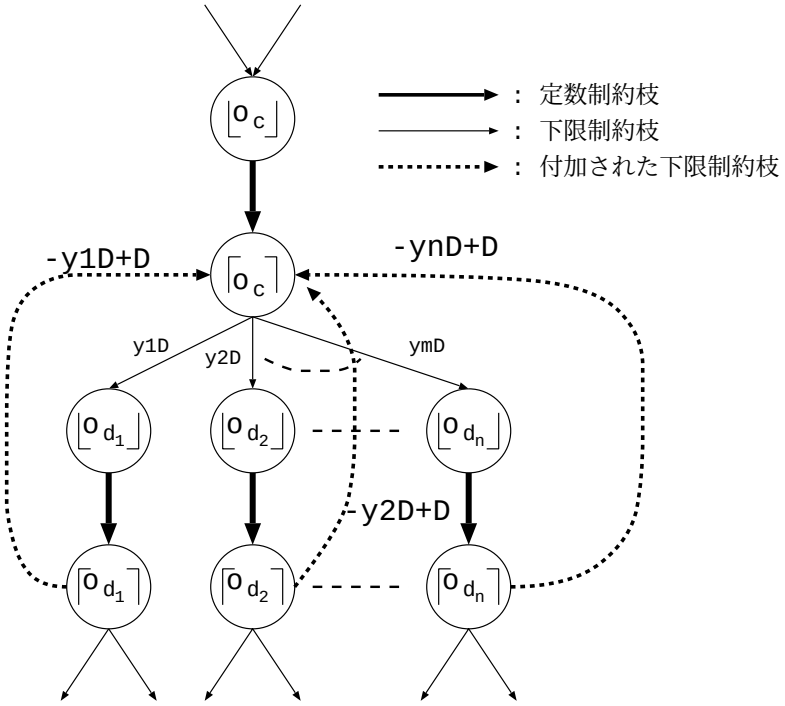


図 5.6: 同一レジスタに割当てられた 1 データに関する制約枝

以上より，問題のインスタンスとして与えられる G, ρ, ξ に対し，

1. G **から初期スケジューリンググラフ** G_{S0} **を生成し，**

2. ρ, ξ **に基づき，資源割り当てからの制約枝を追加し，最終的なパラメトリックスケジューリンググラフ** G_S **を生成する．また** G_S **上の未知変数の集合を** $K \cup R$ **とする．**

この時，パイプラインスケジューリング問題は，

$\Sigma : K \cup R \to Z$ **を決定する問題に帰着される．**

5.2 リタイミングとスケジューリング最適性

先に述べた通り，パイプラインスケジューリングについては，リタイミング $\phi : V_O \to Z$ の自由度がある．初期スケジューリンググラフは，このリタイミングに対応した多数のバリエーションを持ち得る．

先行制約グラフ上でのリタイミング ϕ に対応するスケジューリンググラフ上でのリタイミング $\phi_S : V_{\lceil\rceil} \cup V_{\lfloor\rfloor} \to Z$ は次の様に定められる．

$$\phi_S(\lfloor o_i \rfloor) = \phi_S(\lceil o_i \rceil) = \phi(o_i)$$

なお，リタイミングは枝遅延関数 d の $\tilde{d}$ への変更，但し，$\tilde{d}(p,q) = d(p,q) - \phi(p) + \phi(q)$，とみなすこともできる．

このリタイミングの自由度に関して，次の定理 1 を得ることができる．

定理 1 : $\Sigma : K \cup R \to Z$ を決定する形式の資源割り当て駆動スケジューリングの最適性はリタイミングを考慮した初期スケジューリンググラフの選択に依存しない．

これを証明するために先ず次の補題が導入される．

補題 5 : スケジューリンググラフのみを制約とするスケジューリング問題において，G_S 上のリタイミングはスケジューリングの最適性に影響を与えない．

これはリタイミングの前後において，対応する有向サイクルについての重み（W）総和：遅延（d）総和比が変化しないことから明らか．

次に，ある 1 つの初期スケジューリンググラフを S，S に資源割り当てから要請される制約枝を付加してできるパラメトリックスケジューリンググラフを S_* とする．

一方，S にリタイミング ϕ_S を施してできる初期スケジューリンググラフを $\tilde{S}$，$\tilde{S}$ に制約枝を付加してできるパラメトリックスケジューリンググラフを $\tilde{S}_*$ とする．

この時，S_* と $\tilde{S}_*$ とがリタイミングの関係にあることを示すことができ，補題 5 よりスケジューリングの最適性に関して，S_* と $\tilde{S}_*$ との間に差異がないことが示される．

5.3 K と R の取りうる範囲についての考察

ここでは．与えられた繰り返し周期 Tr の下での未知変数 K_i, R_j の取りうる範囲について考察する．なお，以降では初期スケジューリンググラフが強連結であると仮定する（これ以外については今後の検討課題となっている）．

5.3.1 K の取りうる範囲についての考察 1

$\rho(o_a) = \rho(o_b)$ に起因する制約枝とその未知変数 K_{ab} を考える．式 (5.1),(5.2) より，ある頂点 $\lfloor o_v \rfloor$ を基準点とするスケジューリング $\sigma_{\lfloor o_v \rfloor}$ を考えれば，

$$\begin{aligned} & \frac{-\sigma_{\lfloor o_v \rfloor}(\lfloor o_a \rfloor) + \sigma_{\lfloor o_v \rfloor}(\lceil o_b \rceil)}{Tr} - 1 \\ < \quad & K_{ab} \\ < \quad & \frac{\sigma_{\lfloor o_v \rfloor}(\lfloor o_b \rfloor) - \sigma_{\lfloor o_v \rfloor}(\lceil o_a \rceil)}{Tr} \end{aligned} \tag{5.5}$$

ここで，補題 4 より，

$$\begin{aligned} & \frac{-\sigma_{ALAP\lfloor o_v \rfloor}(\lfloor o_a \rfloor) + \sigma_{ASAP\lfloor o_v \rfloor}(\lceil o_b \rceil)}{Tr} - 1 \\ \leq \quad & \frac{-\sigma_{\lfloor o_v \rfloor}(\lfloor o_a \rfloor) + \sigma_{\lfloor o_v \rfloor}(\lceil o_b \rceil)}{Tr} - 1 \end{aligned} \tag{5.6}$$

及び，

$$\begin{aligned} & \frac{\sigma_{\lfloor o_v \rfloor}(\lfloor o_b \rfloor) - \sigma_{\lfloor o_v \rfloor}(\lceil o_a \rceil)}{Tr} \\ \leq \quad & \frac{\sigma_{ALAP\lfloor o_v \rfloor}(\lfloor o_b \rfloor) - \sigma_{ASAP\lfloor o_v \rfloor}(\lceil o_a \rceil)}{Tr} \end{aligned} \tag{5.7}$$

が成り立つ．また式 (5.5),(5.6),(5.7) は任意の基準点の取り方に対して成り立つことが必要であることから，

$$\begin{aligned} & \underset{o_v \in V_O}{\mathrm{MAX}} \left[\frac{\sigma_{ASAP\lfloor o_v \rfloor}(\lceil o_b \rceil) - \sigma_{ALAP\lfloor o_v \rfloor}(\lfloor o_a \rfloor)}{Tr} - 1 \right] \\ < \quad & K_{ab} \\ < \quad & \underset{o_v \in V_O}{\mathrm{MIN}} \left[\frac{\sigma_{ALAP\lfloor o_v \rfloor}(\lfloor o_b \rfloor) - \sigma_{ASAP\lfloor o_v \rfloor}(\lceil o_a \rceil)}{Tr} \right] \end{aligned} \tag{5.8}$$

と与えられる．

ここで，G_S 上での頂点 a から頂点 b への最長パス長を $L(a,b)$ と記述することにすれば，任意の基準点選択に対して，

$$\begin{aligned}
&\sigma_{ASAP\lfloor o_v\rfloor}(\lceil o_b\rceil) - \sigma_{ALAP\lfloor o_v\rfloor}(\lfloor o_a\rfloor) \\
&= L(\lfloor o_v\rfloor,\ \lceil o_b\rceil) - (-L(\lfloor o_a\rfloor,\ \lfloor o_v\rfloor)) \\
&= L(\lfloor o_a\rfloor,\ \lfloor o_v\rfloor) + L(\lfloor o_v\rfloor,\ \lceil o_b\rceil) \\
&\leq L(\lfloor o_a\rfloor,\ \lceil o_b\rceil) = \sigma_{ASAP\lfloor o_a\rfloor}(\lceil o_b\rceil)
\end{aligned}$$

$$\begin{aligned}
&\sigma_{ALAP\lfloor o_v\rfloor}(\lfloor o_b\rfloor) - \sigma_{ASAP\lfloor o_v\rfloor}(\lceil o_a\rceil) \\
&= -L(\lfloor o_b\rfloor,\ \lfloor o_v\rfloor) - L(\lfloor o_v\rfloor,\ \lceil o_a\rceil) \\
&= -(L(\lfloor o_b\rfloor,\ \lfloor o_v\rfloor) + L(\lfloor o_v\rfloor,\ \lceil o_a\rceil)) \\
&\geq -L(\lfloor o_b\rfloor,\ \lceil o_a\rceil) = -\sigma_{ASAP\lfloor o_b\rfloor}(\lceil o_a\rceil)
\end{aligned}$$

が成り立つ．これらより式 (5.8) は更に，次のように記述できる．

定理 2：

$$\begin{aligned}
\frac{\sigma_{ASAP\lfloor o_a\rfloor}(\lceil o_b\rceil)}{Tr} - 1 &< K_{ab} \\
&< \frac{-\sigma_{ASAP\lfloor o_b\rfloor}(\lceil o_a\rceil)}{Tr}
\end{aligned}$$

なお，ASAP,ALAP はパラメータ値未定の制約枝を無視し，初期スケジューリンググラフの枝とパラメータ値が定まった追加枝のみを考慮して計算するものとしている．

5.3.2 R の取りうる範囲についての考察 1

一方，o_a にて生成され o_{b_x} にて使用されるデータ d_a と，o_f にて生成され o_{g_y} にて使用されるデータ d_f について，$\xi(d_a) = \xi(d_f)$ から生成付加される制約枝上の未知変数 R_{af} についても，式 (5.3),(5.4) より，ある頂点 $\lfloor o_v\rfloor$ を基準点とするスケジューリング $\sigma_{\lfloor o_v\rfloor}$ を考えれば，

$$\begin{aligned}
&\underset{y}{\mathrm{MAX}}\left[t_y - 1 - \frac{-\sigma_{\lfloor o_v\rfloor}(\lceil o_a\rceil) + \sigma_{\lfloor o_v\rfloor}(\lceil o_{g_y}\rceil)}{Tr}\right] \\
&\leq R_{af} \\
&\leq \underset{x}{\mathrm{MAX}}\left[\frac{\sigma_{\lfloor o_v\rfloor}(\lceil o_f\rceil) - \sigma_{\lfloor o_v\rfloor}(\lceil o_{b_x}\rceil)}{Tr} - s_x\right] \qquad (5.9)
\end{aligned}$$

ここで，補題 4 より，

$$\begin{aligned}
&\underset{y}{\mathrm{MAX}}\left[t_y - 1 - \frac{-\sigma_{ALAP\lfloor o_v\rfloor}(\lceil o_a\rceil) + \sigma_{ASAP\lfloor o_v\rfloor}(\lceil o_{g_y}\rceil)}{Tr}\right] \\
&\quad\leq \underset{y}{\mathrm{MAX}}\left[t_y - 1 - \frac{-\sigma_{\lfloor o_v\rfloor}(\lceil o_a\rceil) + \sigma_{\lfloor o_v\rfloor}(\lceil o_{g_y}\rceil)}{Tr}\right] \qquad (5.10)
\end{aligned}$$

及び，

$$\begin{aligned}
&\underset{x}{\mathrm{MAX}}\left[\frac{\sigma_{\lfloor o_v\rfloor}(\lceil o_f\rceil) - \sigma_{\lfloor o_v\rfloor}(\lceil o_{b_x}\rceil)}{Tr} - s_x\right] \\
&\quad\leq \underset{x}{\mathrm{MAX}}\left[\frac{\sigma_{ALAP\lfloor o_v\rfloor}(\lceil o_f\rceil) - \sigma_{ASAP\lfloor o_v\rfloor}(\lceil o_{b_x}\rceil)}{Tr} - s_x\right] \qquad (5.11)
\end{aligned}$$

が成り立つ．また式 (5.9),(5.10),(5.11) は任意の基準点の取り方に対して成り立つことが必要であることから，

$$\begin{aligned}
&\underset{o_v\in V_O}{\mathrm{MAX}}\ \underset{y}{\mathrm{MAX}}\left[t_y - 1 - \frac{\sigma_{ASAP\lfloor o_v\rfloor}(\lceil o_{g_y}\rceil) - \sigma_{ALAP\lfloor o_v\rfloor}(\lceil o_a\rceil)}{Tr}\right] \\
&\quad\leq R_{af} \\
&\quad\leq \underset{o_v\in V_O}{\mathrm{MIN}}\ \underset{x}{\mathrm{MIN}}\left[\frac{\sigma_{ALAP\lfloor o_v\rfloor}(\lceil o_f\rceil) - \sigma_{ASAP\lfloor o_v\rfloor}(\lceil o_{b_x}\rceil)}{Tr} - s_x\right] \qquad (5.12)
\end{aligned}$$

と与えられる．

ここで，G_S 上での頂点 a から頂点 b への最長パス長を $L(a,b)$ と記述することにすれば，任意の基準点選択に対して，

$$\begin{aligned}
&\sigma_{ASAP\lfloor o_v\rfloor}(\lceil o_{g_y}\rceil) - \sigma_{ALAP\lfloor o_v\rfloor}(\lceil o_a\rceil) \\
&\quad= L(\lfloor o_v\rfloor,\ \lceil o_{g_y}\rceil) - (-L(\lceil o_a\rceil,\ \lfloor o_v\rfloor)) \\
&\quad= L(\lceil o_a\rceil,\ \lfloor o_v\rfloor) + L(\lfloor o_v\rfloor,\ \lceil o_{g_y}\rceil) \\
&\quad\leq L(\lceil o_a\rceil,\ \lceil o_{g_y}\rceil) = \sigma_{ASAP\lceil o_a\rceil}(\lceil o_{g_y}\rceil)
\end{aligned}$$

$$\begin{aligned}
&\sigma_{ALAP\lfloor o_v\rfloor}(\lceil o_f\rceil) - \sigma_{ASAP\lfloor o_v\rfloor}(\lceil o_{b_x}\rceil) \\
&\quad= -L(\lceil o_f\rceil,\ \lfloor o_v\rfloor) - L(\lfloor o_v\rfloor,\ \lceil o_{b_x}\rceil) \\
&\quad= -(L(\lceil o_f\rceil,\ \lfloor o_v\rfloor) + L(\lfloor o_v\rfloor,\ \lceil o_{b_x}\rceil)) \\
&\quad\geq -L(\lceil o_f\rceil,\ \lceil o_{b_x}\rceil) = -\sigma_{ASAP\lceil o_f\rceil}(\lceil o_{b_x}\rceil)
\end{aligned}$$

が成り立つ．これらより式 (5.12) は更に，次のように記述できる．

定理 3：

$$\begin{aligned} & \underset{y}{\mathrm{MAX}} \left[t_y - 1 - \frac{\sigma_{ASAP\lceil o_a \rceil}(\lceil o_{g_y} \rceil)}{Tr} \right] \\ \leq \quad & R_{af} \\ \leq \quad & \underset{x}{\mathrm{MIN}} \left[\frac{-\sigma_{ASAP\lceil o_f \rceil}(\lceil o_{b_x} \rceil)}{Tr} - s_x \right] \end{aligned}$$

を得る．

なお，K の時と同様に，ASAP,ALAP はパラメータ値未定の制約枝を無視し，初期スケジューリンググラフの枝とパラメータ値が定まった追加枝のみを考慮して計算するものとしている．

第 6 章

問題の解法

6.1 $\Sigma : K \cup R \to Z$ の厳密解法

先ず，与えられた繰り返し周期 Tr の下でのスケジューリング（周期制約スケジューリング）アルゴリズムを示す．基本的な考えは，K_{ab}, R_{af} の取りうる範囲が与えられた繰り返し周期 Tr の下で一意に定まる場合は，その値を採用することとし，K_{ab}, R_{af} の取りうる範囲が一意に定まらない場合について，全ての値の候補の探索を行う，その時，グラフ中に正のサイクルが存在するか否かを枝刈りに用いた分枝限定法を行うことで探索空間の縮小を行った．

以下にそのアルゴリズムの概要を示す．

6.1.1 厳密解法アルゴリズム

Feasibility (G, Tr)

1. 初期スケジューリンググラフ G_{S0} を作成
2. **if** $(\mathrm{BAB}(G_{S0}, Tr) == 1)$ “FEASIBLE”
   else ”INFEASIBLE”

BAB (G_S, Tr)

1. G_S 上で全点間最大パス長を求める（この時，正のサイクルを検出したら return(0)）
2. $K \cup R$ 中の未定変数について範囲を計算する．
3. **if**（未定変数が残っていない）
   基準点から各頂点への最大パス長を出力して，return(1)
   else if（値の候補がない未定変数が存在する）
   return(0)
   else if（一意に定まる未定変数が存在する）
   一意に定まる未定変数全てについて，値を定め，対応する制約枝を G_S に追加してステップ 1 へ
   else
   1 つの未定変数をとりだし，その値の候補それぞれについて，
   3-1. 値を定め，対応する制約枝を G_S へ追加する．
   3-2. **if** $(\mathrm{BAB}(G_S, Tr) == 1)$ return(1)
   3-3. 3-1. で追加した制約枝を削除する
   return(0)

一方，Tr 最小のスケジューリングを求める問題に対しては，Tr の値を変えながら，Feasibility (G, Tr) を繰り返し実行することとしている．

6.1.2 厳密解法アルゴリズムの計算量

一つの演算器に全ての演算を割り当てる場合，及び一つのレジスタに全てのデータを割り当てる場合，グラフ中の全ての頂点間に制約枝が付加される．頂点数を n とすると，付加される枝数は与えられたグラフの頂点からなる完全グラフの枝数に等しく，およそ n^2 となる．この時，K_{ab}, R_{af} の個数は最大となり，$|K \cup R| = n^2$ で表される．

この時の K_{ab}, R_{af} のそれぞれが取りうる整数値の個数を仮に $f(n)$ とすれば，K_{ab}, R_{af} の取りうる整数値の組合わせの個数は，$f(n)^{|K \cup R|} = f(n)^{n^2}$ で表される．

従って，厳密解法アルゴリズムの計算量は，G_S 上で全点間最大パス長を求めるために，Floyd のアルゴリズムでは，

$$\mathcal{O}(n^3)$$

を要し，それが $f(n)^{|K\cup R|} = f(n)^{n^2}$ 回繰り返されるから，

$$\mathcal{O}(n^3 f(n)^{n^2})$$

となる．

6.2 $\Sigma : K \cup R \to Z$ の heuristics を用いた解法

K_{ab}, R_{af} の取りうる範囲が一意に定まる場合は，厳密解の探索と同様に，その値を採用する．K_{ab}, R_{af} の取りうる範囲が一意に定まらない場合について検討してみる．

ここで，K_{ab}, R_{af} の取りうる範囲について再度考えてみる．

6.2.1 K_{ab} の取りうる範囲についての考察 2

便宜上，定理 2 の不等式を以下の 2 つの不等式に分けて考える．

$$\frac{\sigma_{ASAP\lfloor o_a \rfloor}(\lceil o_b \rceil)}{Tr} - 1 < K_{ab} \tag{6.1}$$

$$K_{ab} < \frac{-\sigma_{ASAP\lfloor o_b \rfloor}(\lceil o_a \rceil)}{Tr} \tag{6.2}$$

ここで定義より，$\sigma_{ASAPi}(j)$ は以下のように記述できる．

$$\sigma_{ASAPi}(j) = \sum_{e \in P_{ij}} W(e) - Tr \sum_{e \in P_{ij}} D_S(r, s)$$

ここで，P_{ij} は i から j までの最長パスを表すものとする．

よって，式 (6.1) の左辺は以下のように示される．

$$\frac{\sum_{e \in P_{\lfloor o_a \rfloor \lceil o_b \rceil}} W(e)}{Tr} - \sum_{e \in P_{\lfloor o_a \rfloor \lceil o_b \rceil}} D_S(e) - 1 \tag{6.3}$$

ここで，$Tr = Tr - \Delta$ とすると（ただし，$0 < \Delta < Tr$），式 (6.3) より式 (6.1) の左辺は増加することがわかる．その増分は以下に示される．

$$\frac{\sum_{e \in P_{\lfloor o_a \rfloor \lceil o_b \rceil}} W(e)}{Tr - \Delta} - \sum_{e \in P_{\lfloor o_a \rfloor \lceil o_b \rceil}} D_S(e) - 1$$

$$
\begin{aligned}
& - \left(\frac{\sum_{e \in P_{\lfloor o_a \rfloor \lceil o_b \rceil}} W(e)}{Tr} - \sum_{e \in P_{\lfloor o_a \rfloor \lceil o_b \rceil}} D_S(e) - 1 \right) \\
= \; & \Delta \left(\frac{\sum_{e \in P_{\lfloor o_a \rfloor \lceil o_b \rceil}} W(e)}{Tr(Tr - \Delta)} \right)
\end{aligned} \tag{6.4}
$$

また，式 (6.2) の右辺についても，式 (6.1) の左辺と同様に以下のように示すことができる．

$$
- \frac{\sum_{e \in P_{\lfloor o_b \rfloor \lceil o_a \rceil}} W(e)}{Tr} + \sum_{e \in P_{\lfloor o_b \rfloor \lceil o_a \rceil}} D_S(e) \tag{6.5}
$$

ここで，$Tr = Tr - \Delta$ とすると（ただし，$0 < \Delta < Tr$），式 (6.5) より式 (6.2) の右辺は減少することがわかる．その減少分は以下に示される．

$$
\begin{aligned}
& - \frac{\sum_{e \in P_{\lfloor o_b \rfloor \lceil o_a \rceil}} W(e)}{Tr} + \sum_{e \in P_{\lfloor o_b \rfloor \lceil o_a \rceil}} D_S(e) \\
& - \left(- \frac{\sum_{e \in P_{\lfloor o_b \rfloor \lceil o_a \rceil}} W(e)}{Tr - \Delta} + \sum_{e \in P_{\lfloor o_b \rfloor \lceil o_a \rceil}} D_S(e) \right) \\
= \; & \Delta \left(\frac{\sum_{e \in P_{\lfloor o_b \rfloor \lceil o_a \rceil}} W(e)}{Tr(Tr - \Delta)} \right)
\end{aligned} \tag{6.6}
$$

従って，式 (6.4)，式 (6.6）より，Tr から Δ だけ Tr を減少させると，K_{ab} の取りうる範囲が縮小することがわかり，逆に，縮小分から Δ を計算で求めることが可能である．

6.2.2 R_{af} の取りうる範囲についての考察 2

便宜上，定理 3 の不等式を以下の 2 つの不等式に分けて考える．

$$
\underset{y}{\mathrm{MAX}} \left[ty - 1 - \frac{\sigma_{ASAP\lceil o_a \rceil}(\lceil o_{g_y} \rceil)}{Tr} \right] \leq R_{af} \tag{6.7}
$$

$$R_{af} \leq \underset{x}{\mathrm{MIN}} \left[\frac{-\sigma_{ASAP\lceil o_f \rceil}(\lceil o_{bx} \rceil)}{Tr} - sx \right] \tag{6.8}$$

式 (6.7) の左辺について，K_{ab} の時と同様に以下のように記述できる．

$$\left\lceil t_y - 1 + \frac{\left(\sum_{e \in P_{\lceil o_a \rceil \lceil o_{gy} \rceil}} W(e) - Tr \sum_{e \in P_{\lceil o_a \rceil \lceil o_{gy} \rceil}} D_S(e) \right)}{Tr} \right\rceil \quad y = 1, 2, \ldots, m$$

さらに，以下のように変形することができ，

$$t_y - 1 + \frac{\sum_{e \in P_{\lceil o_a \rceil \lceil o_{gy} \rceil}} W(e)}{Tr} - \sum_{e \in P_{\lceil o_a \rceil \lceil o_{gy} \rceil}} D_S(e) \quad y = 1, 2, \ldots, m$$

上式より，Tr から Δ だけ Tr を減少させると，式 (6.7) の左辺は増加することがわかる．従って，その増分は以下のように示される．

$$\begin{aligned} & t_y - 1 + \frac{\sum_{e \in P_{\lceil o_a \rceil \lceil o_{gy} \rceil}} W(e)}{Tr - \Delta} - \sum_{e \in P_{\lceil o_a \rceil \lceil o_{gy} \rceil}} D_S(e) \\ - & \left(t_y - 1 + \frac{\sum_{e \in P_{\lceil o_a \rceil \lceil o_{gy} \rceil}} W(e)}{Tr} - \sum_{e \in P_{\lceil o_a \rceil \lceil o_{gy} \rceil}} D_S(e) \right) \\ = & \ \Delta \left(\frac{\sum_{e \in P_{\lceil o_a \rceil \lceil o_{gy} \rceil}} W(e)}{Tr(Tr - \Delta)} \right) \quad y = 1, 2, \ldots, m \end{aligned} \tag{6.9}$$

また同様に式 (6.8) の右辺についても，以下のように記述することができ，

$$\left\lceil -\frac{\sum_{e \in P_{\lceil o_f \rceil \lceil o_{bx} \rceil}} W(e) + Tr \sum_{e \in P_{\lceil o_f \rceil \lceil o_{bx} \rceil}} D_S(e)}{Tr} - s_x \right\rceil \quad x = 1, 2, \ldots, n$$

さらに，以下のように変形できる．

$$\left[-\frac{\sum_{e \in P_{\lceil o_f \rceil \lceil o_{b_x} \rceil}} W(e)}{Tr} + \sum_{e \in P_{\lceil o_f \rceil \lceil o_{b_x} \rceil}} D_S(e) - s_x \right] \quad x = 1, 2, \ldots, n$$

上式より，Tr から Δ だけ Tr を減少させると式 (6.8) の右辺は減少することがわかる．その減少分は以下の式で表される．

$$\begin{aligned} & -\frac{\sum_{e \in P_{\lceil o_f \rceil \lceil o_{b_x} \rceil}} W(e)}{Tr} + \sum_{e \in P_{\lceil o_f \rceil \lceil o_{b_x} \rceil}} D_S(e) - s_x \\ - & \left(-\frac{\sum_{e \in P_{\lceil o_f \rceil \lceil o_{b_x} \rceil}} W(e)}{Tr - \Delta} + \sum_{e \in P_{\lceil o_f \rceil \lceil o_{b_x} \rceil}} D_S(e) - s_x \right) \\ = & \ \Delta \left(\frac{\sum_{e \in P_{\lceil o_f \rceil \lceil o_{b_x} \rceil}} W(e)}{Tr(Tr - \Delta)} \right) \quad x = 1, 2, \ldots, n \end{aligned} \tag{6.10}$$

以上より，Tr から Δ だけ Tr を減少させると，R_{af} の取りうる範囲が縮小することがわかり，逆に，縮小分から Δ を計算で求めることが可能である．

従って，K_{ab}, R_{af} の取りうる範囲が一意に定まらない場合，一意に決定するまで範囲を縮小させると，その縮小分から Δ を計算で求めることが可能となり，求めた Δ の値の大小の順序関係から，K_{ab}, R_{af} の値を決定することが可能である．

ここで，増加分（減少分）を表す式 (6.4）, 式（6.6）, 式（6.9）, 式（6.10) 中で Slack を定義する．

$$\text{Slack} \stackrel{\triangle}{=} \sum_{e \in P_{ij}} W(e)$$

6.2.3 heuristics アルゴリズム

Tr が与えられた時のアルゴリズムの概要は以下の通り

Feasibility (G, Tr)

1. 初期スケジューリンググラフ G_{S0} を作成
2. **if** $(\mathrm{Heu}(G_{S0}, Tr) == 1)$ “FEASIBLE”
   else ”INFEASIBLE”

Heu (G_S, Tr)

1. G_S 上で全点間最大パス長を求める（この時，正のサイクルを検出したら return(0)）
2. $K \cup R$ 中の未定変数について範囲を計算する．
3. **if**（未定変数が残っていない）
   基準点から各頂点への最大パス長を出力して，return(1)
   else if（値の候補がない未定変数が存在する）
   return(0)
   else if（一意に定まる未定変数が存在する）
   一意に定まる未定変数全てについて，値を定め，対応する制約枝を G_S に追加してステップ 1 へ
   else
   全ての未定変数をとりだし，未定変数の個数が一つになるように Δ を計算する
   3-1.**if** 最小の Δ が一意に定まる場合
   3-1-1. 最小の Δ をとる未定変数を候補とする
   3-1-2. 値を定め，対応する制約枝を G_S へ追加して，ステップ 1 へ．
   3-2. **else**
   3-2-1. Δ が同一の未定変数に対して，Slack を計算する
   3-2-2. Slack が最大値をとるものを候補とする．
   3-1-3. 値を定め，対応する制約枝を G_S へ追加して，ステップ 1 へ

厳密解法の場合と同様に，Tr 最小のスケジューリングを求める問題に対しては，Tr の値を変えながら，Feasibility (G, Tr) を繰り返し実行することとしている．

6.2.4 heuristics アルゴリズムの計算量

提案 heuristics アルゴリズムは，$K \cup R$ の未定変数が複数個ある場合，それを一つに決定する．従って，頂点数を n とすると，K_{ab}, R_{af} の個数（n^2）回，全点間最大パス長を求めればよい．故に，提案 heuristics アルゴリズムの計算量は

$$\mathcal{O}(n^3 \times n^2) = \mathcal{O}(n^5)$$

となる．

第 7 章

高位合成への応用

7.1 資源割当て空間探索に基づくデータパス合成

VLSI 微細加工技術の進歩に伴って，配線の VLSI 性能への寄与の度合いが高くなって来ており，モジュール間接続関係やフロアプランを考慮した高位合成（データパス合成）が必要となって来ている．これに対応するためには，高位合成の初期段階からモジュール間接続関係に配慮した最適化が必要と考えられる．こうしたことから，資源割り当て空間探索をコアとする合成手法が検討されている．

ここでは，スケジューリング繰り返し周期 Tr を制約とし，モジュール間結線（端子内結線）本数最小のデータパスを合成する問題を考える．この問題に対し，資源割り当て空間を SA にて探索する方式を採用する．SA の各解の隣接資源割り当て解に対し，スケジューリング繰り返し周期，モジュール間接続本数を評価し，確率的に隣接解を受理しながら資源割り当て解空間を探索するものである．

7.2 実験結果

データパス合成問題への入力インスタンスとして，図 7.1に示した 5 次楕円フィルタを用いた．表 7.1 に厳密解探索実験で用いた制約と得られた結線数をまとめて示す．また，表 7.2 に heuristics アルゴリズムによる探索実験で用いた制約と得られた結線数をまとめて示す．

なお，SA においては隣接解に対し，繰り返し周期制約を満足するもののみを受理候補とし，結線数（FU の左右端子割り当ても含む）を評価して，確率的に解の受理・破棄を決定している．温度は 10 度から 0.01 度まで，倍率 0.98 で下げ，各温度当たり 1000 回の

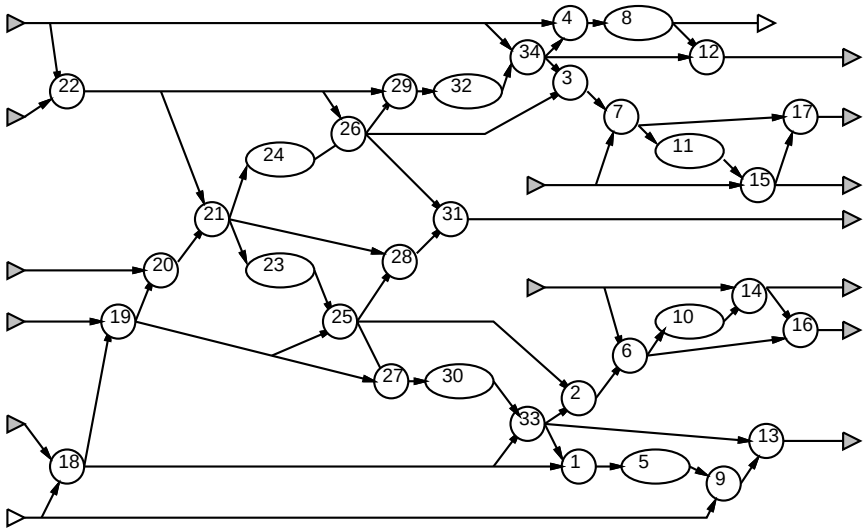


図 7.1: 5 次楕円フィルタ

表 7.1: 厳密解探索実験で用いた制約と得られた結線数

	制約			結果
	演算器	レジスタ	繰り返し周期	結線数
合成 1	$2+, 1\times$	18	22	37
合成 2	$4+, 4\times$	19	16	50

表 7.2: heuristics アルゴリズムによる探索実験で用いた制約と得られた結線数

	制約			結果
	演算器	レジスタ	繰り返し周期	結線数
合成 3	$2+, 1\times$	18	23	39

隣接解評価を行った結果である．計算には SUN ULTRA 5 （動作周波数 400MHz）を使用した．

厳密解の探索においては，計算時間はおよそ 24 時間であり，平均で一解当たり 0.25 sec を要したことになる．また heuristics アルゴリズムを使用した探索においては，計算時間はおよそ 18 時間であり，平均で一解当たり 0.19 sec を要した（なお両実験ともは，FU の左右端子割り当ての最適化（分枝限定法）を含んでいる）．

図 7.2に厳密解の探索で求めた合成 1 の解（アーキテクチャ及びスケジューリング）を，図 7.3に heuristics アルゴリズムによる探索で求めた合成 3 の解（アーキテクチャ及びスケジューリング）を示した．

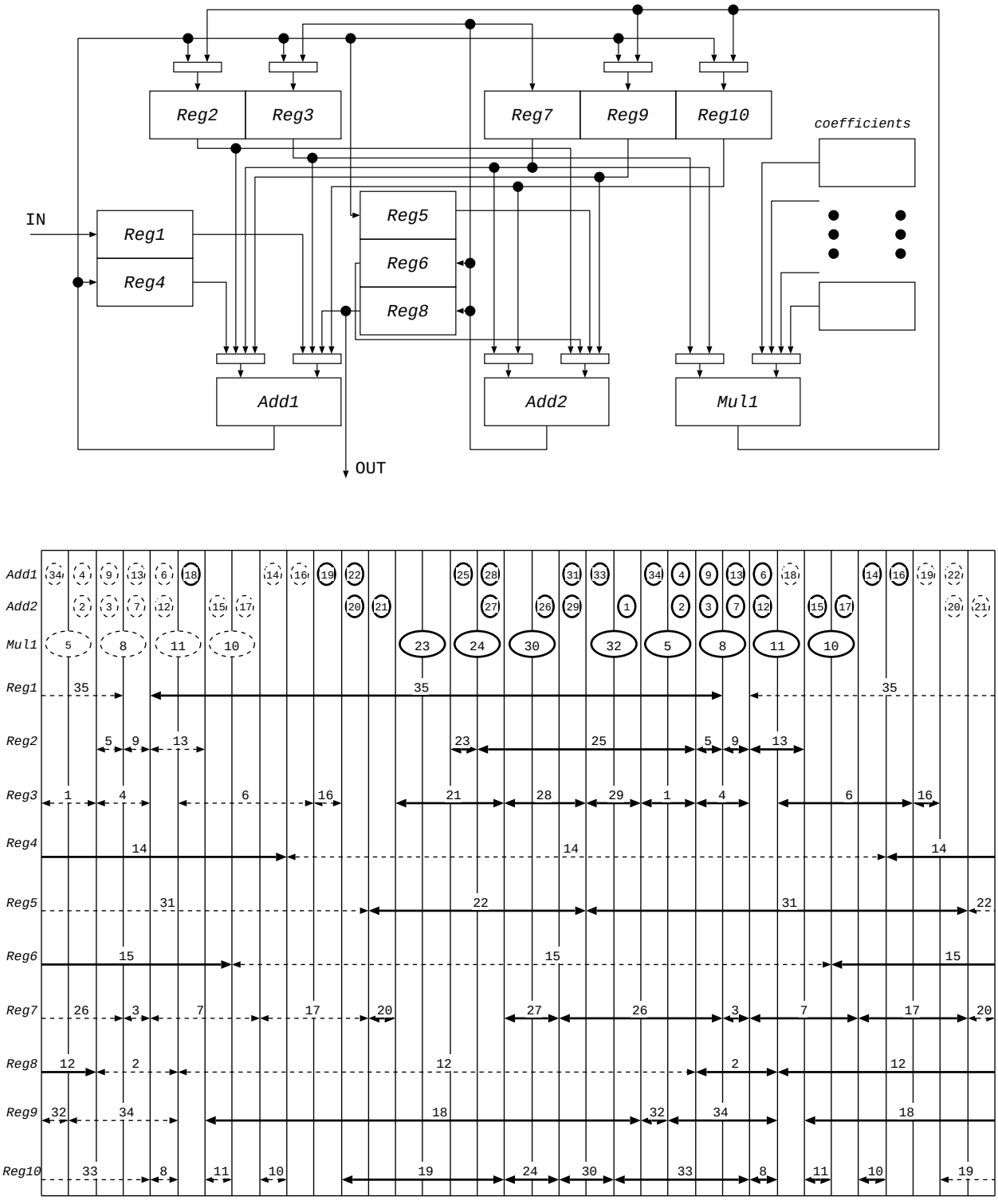


図 7.2: RT レベルアーキテクチャ及びスケジューリング

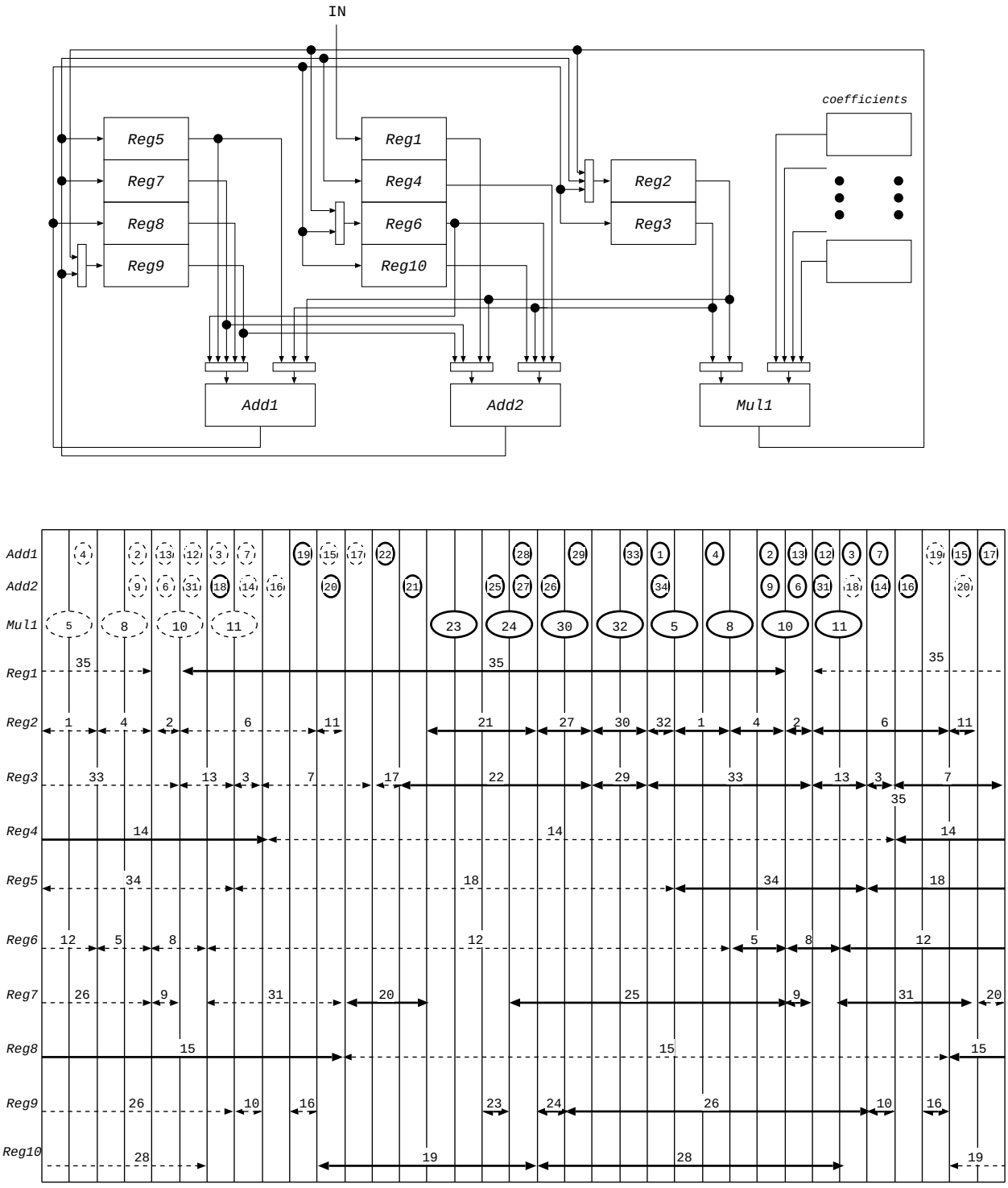


図 7.3: RT レベルアーキテクチャ及びスケジューリング

第 8 章

まとめ

8.1 結論

VLSI 技術の発展による回路素子動作の高速化に伴い，素子間の信号伝達遅延が回路動作時間に占める割合が非常に大きくなっている．そのため信号伝達遅延と関係の深い接続関係あるいはフロアプランの最適性を重視した高位合成手法の開発が必要とされている．このような高位合成手法として資源割り当てを中心とする最適化などが検討されつつある．

本研究では，その手法の一つとして，資源割り当て空間探索をコアとする高位合成を想定し，演算，データの演算器，レジスタへの割り当てが指定された下でのパイプラインスケジューリングについて，制約を反映したパラメトリックスケジューリンググラフを用いた解空間の探索手法を提案した．入力として与えられる先行制約グラフから初期スケジューリンググラフを生成し，さらに資源割り当て指定に基づき制約枝を追加しパラメトリックスケジューリンググラフを生成する．その時，パイプラインスケジューリング問題はパラメトリックスケジューリンググラフ上の未知変数を決定する問題に帰着される．決定すべき変数の範囲計算によって探索空間の縮小と枝刈りを行う分枝限定法により，高速に厳密解の探索を行う手法を提案した．さらに範囲内に複数の変数値をもつ場合の唯一解への変数値の絞りこみに関する heuristcs も考案した．また本研究の提案手法を 5 次楕円フィルタの例を用いて合成実験を行い，提案手法が与えられた資源制約と繰り返し周期の下で，結線数の評価に対して従来得られなかった良好な解を生成することを確認した．これにより，提案手法が 5 次楕円フィルタと同規模の入力に対して，繰り返し周期制約下でモジュール間接続本数最小化のための Simulated Annealing による資源割り当て空間の探索を可能ならしめるものであることを実証した．

8.2 今後の課題

初期スケジューリンググラフが強連結でない場合への拡張及び，さらに効率的な heuristics の開発が今後の課題として残されている．

謝辞

本研究の機会を賜り，御指導，御鞭撻頂きました，北陸先端科学技術大学院大学情報科学研究科 金子 峰雄 助教授ならびに田湯 智 助手に心から感謝の意を表します.

常日頃からお世話になりました北陸先端科学技術大学院大学情報科学研究科博士後期課程 1 年 大橋 功治氏，ならびに北陸先端科学技術大学院大学情報科学研究科・金子研究室の皆様に感謝いたします.

また，勉学・研究の機会を与えてくれた私の家族にも感謝の意を表したいと思います.

参考文献

[1] P.Michel, U.Lauther, P.Duzy, "The synthesis approach to digital system design", Kluwer Academic, 1992.

[2] Y.Nishio, M.Kaneko, S.Tayu, "Assignment Based Approach to High Level Synthesis for Net Relevant Design Creiteria", Technical report of IEICE, VLD98-147, pp.49-56, 1999.

[3] M.Kaneko, Y.Nishio, S.Tayu, "Exact and Heuristic Methods of Assignment Driven Scheduling for Data-Path Synthesis Applications" Proc. ISCAS2000, Vol.II, pp.57-60, 2000.

[4] K.Ito, H.Kunieda, "VLSI System Compiler for Digital Signal Processing: Modulation and Synchronization", IEEE Trans. Circuits Syst., vol.38, No.4, pp.422-433, 1991.

[5] B.Mesman, M.Strik, A.H.Timmer, J.L.van Meerbergen, J.A.G.Jess, "A Constraint Driven Approach to Loop Pipelining and Register Binding", Proc. IEEE DATE, pp.377-383, 1998.

[6] T.Watanabe, N.Ishiura, M.Yamaguch, "A Code Generation Method for Datapath Oriented Codesign of Application Specific DSPs", 電子情報通信学会第 13 回 回路とシステム軽井沢ワークショップ, pp.539-544, 2000.