C-4

準ブール充足可能性判定による TMS320C62x DSP の最適コードスケジューリング

Optimum Code Scheduling for TMS320C62x DSP Using Pseudo Boolean Satisfiability

小林　涼 †　　益井 勇気 †　　石浦 菜岐佐 †
Ryo Kobayashi　Yuuki Masui　Nagisa Ishiura

1　はじめに

VLIW*型 DSP**は, 静的スケジューリングに基づく並列演算により優れた電力性能比を達成するが, 品質の高いコードをいかに生成するかが重要な課題となる. VLIW 型 DSP のコードスケジューリングに関する一般的な研究は多く行われているが, DSP は様々な制約から固有の特殊なデータパス構成を持つものが多く, 具体的な DSP のコードスケジューリングに際しては, これらの制約まで考慮する必要がある. 例えば, [1] は Texas Instrument 社製 TMS320C62x (以下, C62x) を対象に, 演算のクラスタ割り当てを考慮したスケジューリングを行っているが, ユニット毎に許されるオペランドの組み合せやレジスタファイルの容量制約まで考慮した最適化は行っていない. 本稿では, C62x のリニアアセンブラの最適化スケジューラの開発を目標に, 当該 DSP のデータパスに関する制約を考慮したコードスケジューリングを, 準ブール充足可能性判定問題として定式化する.

2　TMS320C62x とリニアアセンブラ

図 1 に C62x のデータパス構成を示す. C62x は 2 つのクラスタ (A, B) を持ち, 各クラスタは 1 つのレジスタファイル (RF) と 4 種類 (L, S, M, D) のユニットから成る. ユニットは合計 8 個あり, 最大で 8 演算を同時に実行できるが, ユニット毎に実行可能な演算が異なる. 例えば, ロード/ストアは D ユニット, 乗算は M ユニットでしか実行できないが, 加算は L, S, D ユニットで実行できる. 演算の実行サイクル数は 1〜6 で, ユニットにかかわらず演算の種類により決まる. レジスタの読み込みは演算の 1 サイクル目, レジスタへの書き込みは 1〜6 サイクル目で行う. 演算に必要なユニットやパスの資源は 1 サイクル目にだけ使用する. 各ユニットのオペランドには基本的に同じクラスタの RF を指定するが, 「クロスパス」を使用すれば反対側のクラスタの RF を 1 サイクル当り最大 1 つ指定できる. クロスパスや即値の使用条件はユニット毎に異なる点に注意を要する. 例えば, L ユニットは両方のオペランドでクロスパスや即値を使用できるのに対し M, S ユニットは第 1 オペランドでしか使用できない. D ユニットは第 1 オペランドにのみ即値を使用できるがクロスパスは使用できない.

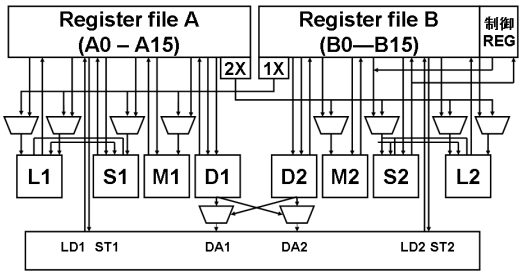
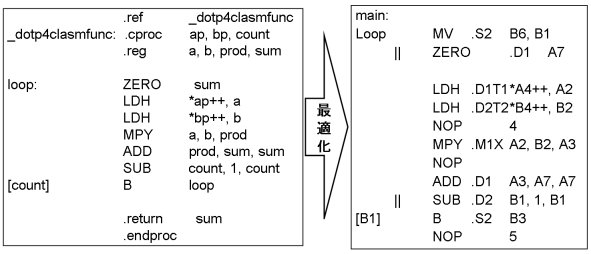


図 1: TMS320C62x のデータパス構成

本研究では C62x のリニアアセンブラの最適スケジューリング問題を扱う. リニアアセンブラは, 図 2(a) のように逐次実行のセマンティクスを持ち, 各命令の実行に必要なユニットやレジスタを指定しないアセンブリ (リニアアセンブリ) から, 図 2(b) の最適化されたアセンブリを出力するものである. コードスケジューラは演算の並列化スケジューリング, 資源割り当て, データ転送演算の挿入を行うが, 演算 (ニーモニック) の変換は行わない.



(a) リニアアセンブリ　　(b) アセンブリ

図 2: リニアアセンブラ

3　コードスケジューリング問題

本稿で扱うコードスケジューリング問題は, 図 3 のように, プログラムの基本ブロック (BB) を表す「依存グラフ」と, プロセッサで実行可能な命令の情報を表す「命令パターン集合」に対し, 実行サイクル数が最小となるように各演算の実行開始サイクルと資源割り当てを決定する問題である. RF の容量を考慮し, 必要なデータ転送演算 (スピル, リロード, レジスタ間転送) の挿入まで含めた最適コードスケジューリングを求める.

BB 中の演算と値を表わす節点の集合をそれぞれ F, V,

†関西学院大学
*Very Long Instruction Word
**Digital Signal Processor

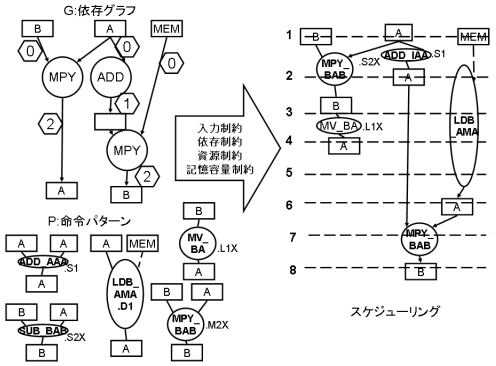


図 3: コードスケジューリング問題

データ依存枝の集合を E とする. 依存グラフ $G = (V \cup F, E)$ は非巡回の有向 2 部グラフ ($E \subseteq V \times F \cup F \times V$) である. 各枝 $e \in E$ にはオペランド番号 $o(e)$ と整数値の遅延 $d(e)$ が定義されている. データを格納する記憶要素の集合を M とし, $m \in M$ の容量を $C(m)$ とする. C62x では $M = \{A, B, MM\}$ であり[†], $C(A) = C(B) = 14$[‡], $C(MM) = \infty$ とする. $V_I \subseteq V$ および $V_O \subseteq V$ はそれぞれ BB の入力値, 出力値の集合であり, 入力値には記憶要素 $i(v) \in M$ が定義されている. t_{max} は実行サイクル数の上限, $T = \{1, 2, \cdots, t_{max}\}$ は実行サイクルの集合を表す.

命令パターン集合 P の要素 p には, p が使用するユニットの集合 $U(p)$, パスの集合 $X(p)$, 遅延 $d(p)$ が定義されている. 演算 $f \in F$ の結果を記憶要素 m に格納する命令パターンの集合を $P_{f,m}$, 命令パターン p において i 番目のオペランドの値を格納する記憶要素を $m(p, i)$ とする. S を転送演算の集合, $d(s)$ を転送演算 s の遅延とする. $P_{s,n,m}$ を転送演算 s で記憶要素 n から記憶要素 m へデータを転送する命令パターンの集合とする.

変数 $\alpha_{t,v,m}$ は, サイクル t に節点 v が記憶要素 m に存在すれば 1, そうでなければ 0 をとる. $\xi_{t,f,p}$ はサイクル t に節点 f の演算を命令パターン p で実行すれば 1, そうでなければ 0 をとる. $\tau_{t,v,p}$ はサイクル t に節点 v を命令パターン p で転送すれば 1, そうでなければ 0 をとる.

制約条件は以下の 4 つである.

(1) 入力制約:

$$\begin{cases} \forall v \in V_I : \quad \alpha_{0,v,m} = 1 \quad \text{iff} \quad i(v) = m. \\ \forall v \in V - V_I, \forall m \in M : \quad \alpha_{0,v,m} = 0. \end{cases}$$

(2) 依存制約:

1. $\forall t \in T, \forall (f, v) \in E, \forall s \in S, \forall n \in M, \forall m \in M$:

$$\alpha_{t,v,m} \rightarrow \alpha_{t-1,v,m} \vee \Big(\bigvee_{p \in P_{f,m}} \xi_{t-d(f,v),f,p} \Big) \vee \Big(\bigvee_{p \in P_{s,n,m}} \tau_{t-d(s),v,p} \Big).$$

2. $\forall t \in T, \forall f \in F, \forall p \in P_{f,m}$:

$$\xi_{t,f,p} \rightarrow \bigwedge_{(v,f) \in E} \alpha_{t-d(v,f),v,m(p,o(e))}.$$

3. $\forall t \in T, \forall v \in V, \forall s \in S, \forall m, n \in M, \forall p \in P_{s,n,m}$:

$$\tau_{t,v,p} \rightarrow \alpha_{t,v,n}.$$

(3) 資源制約: サイクル t でユニットまたはパス r を使用する変数の集合を $B_r^t = \{\xi_{t,f,p}, \tau_{t,v,p} \mid t \in T, f \in F, U(p) = r \vee X(p) = r\}$ とする.

$$\forall t \in T, \forall r \in U(p) \cup X(p) : \quad \sum_{\beta \in B_r^t} \beta \leq 1.$$

(4) 記憶容量制約:

$$\forall t \in T, \forall m \in M : \quad \sum_{v \in V} \alpha_{t,v,m} \leq C(m).$$

以上の制約を満たす $\alpha_{t,v,m}, \xi_{t,f,p}, \tau_{t,v,p}$ への割り当てを求めることにより, 実行可能なスケジューリングが得られる. 実行サイクル数の最小化は以下のように表せる.

$$\text{maximize} \quad \sum_{t \in T} \Big(\bigwedge_{v \in V_O} \Big(\bigvee_{m \in M} \alpha_{t,v,m} \Big) \Big).$$

4 実験結果

上記の定式化に基づき, 最適スケジューリングを求める処理系を準ブール充足可能性判定のソルバ PBS Ver2.1 for Win [2] を用いて実装した. 実験結果を表 1 に示す. 「CPU」は PBS が解を得るのに要した時間である[§]. 演算節点 50 程度からなる DFG に対して最適スケジューリングを数秒で得られた.

表 1: 実験結果

プログラム	演算節点	値節点	制約式の数	最適サイクル数	CPU(sec)
fir	49	59	41328	17	3.0
ellip	34	43	37062	17	2.2
convolution	31	63	34543	16	29.8

5 むすび

本稿ではクロスパス, ユニット毎のオペランド組合せ, スピル/リロードを考慮に入れた C62x の最適コードスケジューリングの手法を提案した. 多出力 (複数のレジスタを更新する演算) への対応が今後の課題である.

参考文献

[1] R. Leupers: "Instruction Scheduling for Clustered VLIW DSPs," in *Proc. IPACT 2000*, pp. 291–300 (Oct. 2000).

[2] F. A. Aloul et al.:"Generic ILP versus Specialized 0-1 ILP: An Update," in *Proc. ICCAD 2002*, pp. 450–457 (Nov. 2002).

[†] A, B はそれぞれ RF A, RF B, MM は主記憶を表す.

[‡] RF の容量は 16 だが, うち 2 つは用途が予約されているとする.

[§] Cygwin ver.2.05b.0, Intel Celeron 2.6GHz, メモリ 768MB の環境.