一般同期回路における動作周波数向上のための遅延挿入手法に関する研究

A Study on Delay Insertion Method to Improve Clock Period of General-Synchronous Circuit

電気電子情報通信工学専攻　新井　祐樹
Yuki ARAI

1. 序論

完全同期式（complete-synchronous framework）回路ではクロック信号が全てのレジスタへ同時に供給されるという仮定の下で設計されるため，セットアップ制約を満たすには，クロックの周期は組み合わせ回路の最大遅延（クリティカル遅延）より長くなければならない[1]．しかし，各レジスタに個別のタイミングでクロック信号を供給できるなら，クロック周期をクリティカル遅延より小さくできる．そのようなクロック分配手法の回路は一般同期式（general-synchronous framework）と呼ばれ，これまでにいくつかの研究がなされている[2-4].

一般同期式回路のクロック周期を短くする手法の一つに，組み合わせ回路の最小遅延を遅延挿入によって増加させる方法がある．この方法によって達成可能な最小周期と最小遅延挿入量は混合整数計画法を用いて求めることができるが[5]，大規模な回路に対して適用することが実用上困難なため，いくつかの発見的手法が提案されている[6-8]．これらは制約グラフ上の閉路に注目するが，制約グラフの全ての閉路を調べることは困難なため，手法[8]では二分木探索と Bellman-Ford 法を用いて最小動作周期を求めている．

組み合わせ回路の最小遅延経路に遅延挿入を行う手法の内，手法[9]は，クリティカル遅延を超えないように遅延挿入するため，最小周期を達成できないこともある．これに対して，手法[4]は，クリティカル遅延を超えて遅延挿入し，最適な遅延挿入方法を探索しているが，与えられた動作周期で定められるタイミングを使い，フローグラフの最小カットを必要とする．

一方，手法[10]は，制約グラフの点や枝の削除を繰り返すことにより，動作可能な最小周期と遅延挿入すべき閉路を見出すと共に，製造ばらつき等に起因する遅延ばらつきも考慮しており，点や枝の削除の際に用いられる演算は，統計的静的タイミング解析[11]と同様，加算と最大値演算だけである．従って，この手法は遅延ばらつきを考慮した遅延挿入手法を構築する上で，有用な手法となりうる．しかし，重要なパラメータであるクロック周期の決定法や枝削除の方法の適切性が明確でない．そのため，これらが不適切な場合，性能が大きく劣化してしまう．点や枝の削減を繰り返す（グラフ縮小）技法を用いた統計的な遅延挿入手法を構築するためには，これらと性能の関係や，適切な決定法を見いだす必要がある．

そこで，本文では，グラフ縮小技法[10]を用いた新しい発見的遅延挿入手法を提案すると共に，その性能を評価する．クロック周期や枝削除方法の影響を分かりやすく評価するため，ここでは全ての遅延を定数として扱い，グラフ縮小技法が統計的遅延挿入手法に利用可能か否かの調査を行う．

2. 準備

一般同期式回路において，レジスタ r にクロック信号が到達するタイミング，すなわちクロックドライバから r までの遅延時間を $S(r)$ と書き，$d_{max}(a,b)$ および $d_{min}(a,b)$ をそれぞれレジスタ a からレジスタ b への最大遅延および最小遅延とする．このとき，セットアップ制約およびホールド制約は次のように書ける．

・セットアップ制約:

$$S(a) + d_{max}(a,b) \le T_{clk} + S(b), \quad (1)$$

・ホールド制約:

$$S(b) \le S(a) + d_{min}(a,b), \quad (2)$$

ここで T_{clk} は与えられたクロック周期である．

与えられた順序回路 SC の制約グラフ G_{SC} とは，点が SC のレジスタに対応し，信号が伝搬可能なレジスタに対応した点間に，上記の制約に対応した有向枝を付加した有向グラフである．すなわち，レジスタ a からレジスタ b へ信号が伝播される際，点 a から点 b への有向枝 $e_{\mathrm{s}} = (a,b)$ はセットアップ枝と呼ばれ，重み $w(e_{\mathrm{s}}) = d_{max}(a,b) - T_{clk}$ を持ち，点 b から点 a への有向枝 $e_{\mathrm{h}} = (b,a)$ はホールド枝と呼ばれ，重み $w(e_{\mathrm{h}}) = -d_{min}(a,b)$ を持つ．この制約グラフ G_{SC} おいて，上記の 2 つの制約を満たすように各点へ重み $S(r)$ を割り当てることができれば，各レジスタ r に $S(r)$ のタイミングでクロック信号を伝達することにより，回路 SC をクロック周期 T_{clk} で動作させることができる．

今，SC にレジスタ a からレジスタ b へ k 個のレジスタ $r_i\,(1 \le i \le k)$ を通る経路 $\mathrm{P} = (a = r_0, r_1, r_2, \cdots, r_k = b)$ があるとき，経路 P に対するセットアップ制約およびホールド制約はそれぞれ次のように書ける．

$$S(a) + \sum_{i=1}^{k} d_{max}(r_{i-1}, r_i) - k \cdot T_{clk} \le S(b), \quad (3)$$

$$S(b) \le S(a) + \sum_{i=1}^{k} d_{min}(r_{i-1}, r_i). \quad (4)$$

従って，SC 上の閉路 $\mathrm{C} = (a = r_0, r_1, r_2, \cdots, r_k = a)$ に対して次式を得る．

$$\sum_{i=1}^{k} d_{max}(r_{i-1}, r_i) - k \cdot T_{clk} \le 0, \quad (5)$$

$$-\sum_{i=1}^{k} d_{min}(r_{i-1}, r_i) \le 0, \quad (6)$$

これらより，制約グラフ G_{SC} 内の全ての閉路の重みの総和は非正でなければならないことが分かる．しかし，T_{clk} の値が小さいと，枝の重みの総和 $w(\mathrm{C}) = \sum_{e \in \mathrm{C}} w(e)$が正の閉路 C が生じる．そのような場合，各レジスタ r に対する適切なクロックタイミング $S(r)$ を見つけることができず，クロック周期 T_{clk} で動作させることができない[6]．また，式(5)より，一般同期式回路の最小（クロック）周期 T_{Min} が次式で定められることが分かる[4].

$$T_{Min} = \mathrm{Max}\left[\frac{D_{\mathrm{S}}(\mathrm{C})}{N_{\mathrm{S}}(\mathrm{C})} \mid \mathrm{C} \text{ は順序回路 SC の閉路} \right], \quad (7)$$

ここで，$N_{\mathrm{S}}(\mathrm{C}) = k$ および $D_{\mathrm{S}}(\mathrm{C}) = \sum_{i=1}^{k} d_{max}(r_{i-1}, r_i)$ は，それぞれ閉路 C 上のレジスタの個数および最大遅延の合計である．制約グラフ内の閉路 C’が順序回路 SC 内の閉路 C に対応するならば，$N_{\mathrm{S}}(\mathrm{C})$ は閉路 C’上のセットアップ枝の個数であり，$D_{\mathrm{S}}(\mathrm{C})$ は，

$$D_{\mathrm{S}}(\mathrm{C}) = \sum_{e \in \mathrm{C_S}} w(e) + N_{\mathrm{S}}(\mathrm{C}) \cdot T_{clk}, \quad (8)$$

で与えられる．ここで，$\mathrm{C_S}$ は閉路 C’ 内のセットアップ枝の集合である．

回路 SC において，レジスタ a からレジスタ b に 2 つの再収斂経路 $\mathrm{P_1}$ および $\mathrm{P_2}$ があると，制約グラフ上にはこれらの経路に対応した閉路 C が生成され，この閉路はセットアップ枝とホールド枝を含む．そこで，$\mathrm{C_S}$ および $\mathrm{C_H}$ をそれぞれ C 上のセットアップ枝およびホールド枝の集合とし，これらの重みの総和をそれぞれ $D_{\mathrm{S}}(\mathrm{C})$ および $D_{\mathrm{H}}(\mathrm{C})$ と書く．

$$D_{\mathrm{S}}(\mathrm{C}) = \sum_{e_{\mathrm{s}} \in \mathrm{C_S}} \{w(e_{\mathrm{s}}) + T_{clk}\} = \sum_{e_{\mathrm{s}} \in \mathrm{C_S}} d_{max}(e_{\mathrm{s}}), \quad (9)$$

$$D_{\mathrm{H}}(\mathrm{C}) = \sum_{e_{\mathrm{h}} \in \mathrm{C_H}} |w(e_{\mathrm{h}})| = \sum_{e_{\mathrm{h}} \in \mathrm{C_H}} d_{min}(e_{\mathrm{h}}), \quad (10)$$

ここで，$e_{\mathrm{s}} = (x, y) \in \mathrm{C_S}$ に対して $d_{max}(e_{\mathrm{s}}) = d_{max}(x, y)$ であり，$e_{\mathrm{h}} = (y, x) \in \mathrm{C_H}$ に対して$d_{min}(e_{\mathrm{h}}) = d_{min}(x, y)$ である．従って，C の重みの総和は次式で与えられる．

$$w(\mathrm{C}) = D_{\mathrm{S}}(\mathrm{C}) - N_{\mathrm{S}}(\mathrm{C}) \cdot T_{clk} - D_{\mathrm{H}}(\mathrm{C}). \quad (11)$$

G_{SC} 上の閉路 C に対して，$w(\mathrm{C}) \le 0$ するには，クロック周期 T_{clk} は次式を満たさなければならない．

$$\mathrm{E_S[C]} = \frac{D_{\mathrm{S}}(\mathrm{C}) - D_{\mathrm{H}}(\mathrm{C})}{N_{\mathrm{S}}(\mathrm{C})} \le T_{clk}. \quad (12)$$

以下では，式(12)の左辺を閉路 C の S 平均と呼び $\mathrm{E_S[C]}$ と記す．また，$\mathrm{E_S[C]}$ が最小周期 T_{Min} より大きい閉路をクリティカル閉路と呼ぶ．G_{SC} 内にクリティカル閉路がなければ，回路 SC を最小周期 T_{Min} のクロック周期で動作させることができる．

式(12)を満たすには T_{clk} を大きくすればよいが，それ以外に S 平均 $\mathrm{E_S[C]}$ を小さくしてもよい．これを行うには，$D_{\mathrm{H}}(\mathrm{C})$ を大きくし，閉路 C 上のホールド枝 $e_h = (b, a)$ の重み $w(e_{\mathrm{h}})$ を小さくする，すなわちレジスタ a からレジスタ b への最小遅延経路に遅延素子を挿入し，最小遅延 $d_{min}(a, b)$ を大きくすればよい[4,5,9]．この方法により，全閉路の S 平均を最小周期 T_{Min} 以下にできれば，回路 SC を最小周期 T_{Min} のクロックで動作させることができる．

その際，$d_{min}(a, b)$ の値を $d_{max}(a, b)$ を超えて大きくしなければ，最小周期 T_{Min} の値は変化しないが，クリティカル閉路を消去できない，すなわち，全閉路の S 平均を T_{Min} 以下にできないことがある．また，クリティカル閉路 C のホールド枝 $e_h = (b, a)$ に対応する最小遅延 $d_{min}(a, b)$ を $d_{max}(a, b)$ を超えて増加させることによって C の S 平均 $\mathrm{E_S[C]}$ を T_{Min} 以下にできることもある．

3. 提案手法

この節では提案手法を示す．本研究では，回路の構造を考えず制約グラフ G_{SC} を直接扱い，組み合わせ回路内に自由に遅延素子を挿入できるものとする．この仮定は，グラフ縮小技法の性能評価が本文の主目的であるからである．

Step 0° <初期化>: G_{SC} から全ての自己閉路を取り除く．このとき，自己閉路はセットアップ枝であり，回路 SC において一つのレジスタから同じレジスタに戻る経路に対応する．次に，最小周期 T_{Min} を求め，$T_{clk} := T_{Min}$ とする．以下では，Step 1° で見つかる最大の S 平均を LS_{mean}と書く．これらの初期値は $LS_{mean} = \infty$ とする．

Step 1° <グラフ縮小>: 与えられた G_{SC} と枝重み $w(\cdot)$ に対して，点削除（serial merge[10]）および枝削除（parallel merge[10]）を繰り返し，自己閉路が見出される度に，G_{SC} 内の閉路のリストに挿入し，最後にこのリストを閉路リストとして出力する．この閉路リストを用いて LS_{mean} を計算する．

Step 2° <終了判定>: $LS_{mean} = T_{Min}$ であれば終了する．そうでなければ，S 平均が LS_{mean} である閉路 $\mathrm{C}_{LS_{mean}}$ がホールド枝を持つか否かを調べ，持っていなければ終了する．この場合，クロック周期を T_{Min} にできなかったことになる．これら以外の場合，Step 3° に行き，遅延挿入を行う．

Step 3° <遅延挿入>: ホールド枝を 1 本しか持たないクリティカル閉路 C が存在する場合，そのホールド枝を必須枝と呼び，C に対して以下の式で定義される必要量 Δ を求め，

$$\Delta = N_{\mathrm{S}}(\mathrm{C}) \cdot \{\mathrm{E_S[C]} - T_{Min}\}, \quad (16)$$

必須枝の重みの絶対値を Δ だけ増加する．すなわち，必須枝 $e_{\mathrm{h}} = (b, a)$ の最小遅延 $d_{min}(a, b)$ を遅延挿入により Δ だけ増加する．これにより，C の S 平均 $\mathrm{E_S[C]}$ を最小周期 T_{Min} まで削減できる．どのクリティカル閉路もホールド枝を複数本持つと

き, S 平均が最大の閉路 $C_{LS_{mean}}$ の中のホールド枝を 1 本選び, 遅延挿入によってその重みを増やす.

Step 4° <更新>: $T_{clk} := LS_{mean}$ として，Step 1° に戻る.

3.1. グラフ縮小

Step 1° では，G_{SC} 内の各セットアップ枝に対して $N_S(e) := 1$，各ホールド枝に対して $N_S(e) := 0$ とした後，G_{SC} の点が 1 個になるまで，点削除と枝削除を繰り返す．この反復によって自己閉路が現れたらそれを G_{SC}から取り除き, その重みと共に閉路リストへ加える. なぜなら, これらの自己閉路は初期の G_{SC} 内の閉路 C に対応し, その S 平均は次式で計算できるからである.

$$E_S[C] = \frac{w(C)}{N_S(C)} + T_{clk}. \quad (15)$$

点削除は，$|In(v)| \cdot |Out(v)| - \{|In(v)| + |Out(v)|\}$ の値が最小の点 v に対して行う. ここで, $In(v)$ は点 v の入力枝 (x, v) のソース点 x の集合であり，$Out(v)$ は出力枝 (v, y) のシンク点 y の集合である. 点 v の削除は，各点対 $(x, y) \in In(v) \times Out(v)$ に対して，2 つの枝 $e_{in} = (x, v)$，$e_{out} = (v, y)$ を新たな枝 $e = (x, y)$ で置き換えることにより行う．従って，$|In(v)| \cdot |Out(v)| - \{|In(v)| + |Out(v)|\}$ はこの操作で増加する枝の本数である.

新しい枝 $e = (x, y)$ の $N_S(e)$ および $w(e)$ はそれぞれ $N_S(e_{in}) + N_S(e_{out})$ および $w(e_{in}) + w(e_{out})$ とする．また，$x = y$ ならば，e は自己閉路であるが，そのような場合，枝 e を G_{SC} に付加せず，対応する閉路を閉路リストに加える.

点削除で増加した枝の本数を減らすため，並列枝の削除を行い，重みの大きい枝だけを残す．その際，閉路リスト中の閉路の最大の S 平均 LS_{mean} が，G_{SC} の全閉路の中で最大の S 平均 S_{max} と等しくなるように枝を残したいが, セットアップ枝 e_s の重み $w(e_s)$ は T_{clk} の値によって変動するため，容易ではない.

文献[10]では, 新しい閉路が見つかる度に, その S 平均とその時点の T_{clk} との統計的最大値演算をした結果を新たな T_{clk} としている. すなわち, 定数の場合であれば, 大きい方を新たな T_{clk} としている. また, 重みが他の枝より小さくなる確率が閾値（0.99）以上，すなわち明らかに重みの違う枝の場合のみ削除を行う．このような方法では，G_{SC} の全閉路の中で最大の S 平均 S_{max} を見いだせる保証はない.

今，2 本の並列枝 $e_a = (x, y)$，$e_b = (x, y)$ と y から xへ経路 P が存在し，これらの重みがそれぞれ $w(e_a)$，$w(e_b)$，$w(P)$ であるとき，$w(e_a) > w(e_b)$ ならば枝 e_b が削除される．このとき，枝 e_a と経路 P で構成される閉路 C_a の S 平均 $E_S[C_a]$ が，e_b と P で構成される閉路 C_b の S 平均 $E_S[C_b]$ より小さいならば，大きい S 平均 $E_S[C_b]$ の情報が失われてしまう．しかし，T_{clk} が $E_S[C_a]$ 以上になれば，$w(e_b) \geq -w(P) \geq w(e_a)$ が成り立つ（ただし$T_{clk} \leq E_S[C_b]$でなければならない）．すなわち，e_b が削除されたとしても，$E_S[C_a]$ が残っていれば，Step 4° の更新手続きから分かるように，次の反復において T_{clk} が $E_S[C_a]$ 以上になる可能性は高く，次の反復では e_b が残り，$E_S[C_b]$ を見いだされる可能性がある．そのため，提案手法では Step 4° で $T_{clk} := LS_{mean}$ とし, グラフ縮小中に T_{clk} の値を変更しない.

3.2. 遅延挿入

Step 3°では, まず必須枝を持つ各クリティカル閉路 C に対して，C の S 平均 $E_S[C]$ を最小周期 T_{Min} にするのに必要な遅延挿入量（必要量）Δ だけ遅延挿入している．この方法（技法 Essential と呼ぶ）により，1 度に必要かつ多量の遅延を複数のホールド枝へ挿入するできるため，反復回数を減らすことができる．しかし，2 節で述べたように，他のクリティカル閉路の S 平均を増加させてしまう可能性もある. そこで, 実験によってその効果を調査する.

必須枝を持つクリティカル閉路が存在しないならば, S 平均が最大の閉路 $C_{LS_{mean}}$ の中からホールド枝 $e_h = (b, a)$ を 1 つ選び，その最小遅延 $d_{min}(a, b)$ を少しずつ増加させる．なぜなら，$C_{LS_{mean}}$ が 3 本のホールド枝を持つならば, 必要量 Δ を $\Delta = \Delta_1 + \Delta_2 + \Delta_3$ と分割し, 各ホールド枝にこれらの量を分配して遅延挿入すれば, $C_{LS_{mean}}$ の S 平均を T_{Min} にすることもできるし，他の閉路の S 平均を悪化させる危険性も少ないからである.

選択するホールド枝 $e_h = (b, a)$ は，反復回数を削減するため，多くのクリティカル閉路にホールド枝として含まれ，かつ遅延挿入によってクリティカル閉路になるようなセットアップ枝 $e_s = (a, b)$ を含む閉路が少ないものが好ましい. しかし, 各ホールド枝に対応するセットアップ枝を含む閉路を特定し, 式(13)をチェックすることは困難である．そこで，e_h をホールド枝として持つクリティカル閉路が最も多く Step 1° で生成された閉路リストに含まれるものを選び，それに遅延挿入する．その量は

$$\delta = \mathrm{Min}[\,\Delta,\ \mathrm{IR} \cdot LS_{mean}\,]. \quad (16)$$

とする．ここで，IR はホールド枝の遅延を少しずつ増やすために導入した改善率である.

4. 実験結果

グラフ縮小技法を評価するため，提案手法を C 言語でプログラムし，3.50GHz Intel Xeon Processor E3-1241 v3，16GByte RAM のパソコンを用いて，ISCAS89 ベンチマーク回路に適用した．ただし，実験では，最大遅延を超える遅延挿入の効果を調べるため，各枝 $e = (a, b)$ の最小遅延を増加させ，$d_{min}(a, b) = d_{max}(a, b)$ としてから提案手法を実行した. そのため, 挿入した遅延の総量は必要以上に大きくなっている.

表 1 の D_{add}，*ite*，および CPU は，それぞれ挿入され

た遅延の総量，反復回数，および CPU 時間である．各回路に対して，改善率 IR を 0.1 から 0.4 へ変化させ，技法 Essential を用いなかった場合の結果を下段に示す．これらより，s13207 と s38417 以外の回路では，技法 Essential を用いた提案手法の方が良い結果であることが分かる．s13207 と s38417 では，必須枝を持つクリティカル閉路が少ないため，Essential の効果が発揮されず，必須枝を持つクリティカル閉路を検索するための時間は必要であったため，CPU 時間が増加した．

いくつかの例外を除き，IR が増えるに連れ，反復回数と CPU 時間は減少し，遅延の総量 D_{add} が増加している．従って，IR は 0.2 から 0.3 の間が適切であると言える．

5. むすび

本文では，一般同期式回路に対する遅延挿入にグラフ縮小技法を用いた新しい発見的手法を提案し，その評価を行った．グラフ縮小技法は，点と枝の削減を行う際，遅延量に対して加算と最大値演算しか実行しないため，統計的手法に変更し易い．

実験結果から，提案手法の有効性が見いだせたので，今後の課題は統計的遅延挿入手法の確立である．

文　献

[1] D.D. Gajski, Principles of Digital Design, Prentice Hall, 1997.

[2] J.P. Fishburn, “Clock skew optimization,” IEEE Trans. on Computers, vol.39, no.7, pp.945-951, 1990.

[3] R. Mader, E. Friedman, A. Litman, and I. Kourtev, “Large scale clock skew scheduling techniques for improved reliability of degital synchronous VLSI circuits,” Proc. Int. Symp. on Circuits and Systems, pp.357-360, 2002.

[4] Y. Kohira, S. Tani, and A, Takahashi. “Minimization of delay insertion in clock period improvement in general synchronous framework,” IEICE Trans. on Fundamentals, vol.E92-A, no.4, pp.1106-1114, 2009.

[5] B. Taskin, and I. S. Kourtev, “Delay insertion method in clock scheduling,” IEEE Trans. on Computer-Aided Design, vol.25, no.4 pp.651-663, 2006.

[6] R.B. Deokar and S.S. Sapatnekar, “A praph-theoretic approach to clock skew optimization,” Proc. Int. Symp. on Circuits and Systems, pp.407-410, 1994.

[7] C. Albrecht, B. Korte, J. Schietke, and J. Vygen, “Cycle time and slack optimization for VLSI-chips,” Proc. Int. Conf. on Computer-Aided Design, pp.233-238, 1999.

[8] T. Ishida, Y. Kohira, and A. Takahashi, “Performance evaluation of negative cycle detection algorithms,” IPSJ SIG Tech. Report, 2006-AL-107, pp.45-50, 2004.

[9] N. Shenoy, R. Brayton, and A. Sangiovanni- Vincentelli, “Minimum padding to satisfy short path constraints,” Digests of Int. Conf. on Computer-Aided Design, pp. 156–161, 1993.

[10] B. Li, N. Chen, and U. Schlichtmann, “Fast SSTA for circuits with post-silicon tunable clock buffers,” Digests of Int. Conf. on Computer-Aided Design, pp.111-117, 2011.

[11] D.Blaauw, K.Chopra, A.Srivastave, L.Scheffer, “Statistical timing analysis: From basic principles to state of the art,” IEEE Trans. CAD/ICAS, vol.27, no.4, pp.589-607, 2008.

表 1: 挿入遅延の総和，繰り返し回数，および CPU 時間．(下段，技法 Essential 無し)

IR	0.1		0.2		0.3		0.4	
SC	D_{add}[μs]	*ite*, (CPU[s])	D_{add}[μs]	*ite*, (CPU[s])	D_{add}[μs]	*ite*, (CPU[s])	D_{add}[μs]	*ite*, (CPU[s])
s208	1.42	362, (0.44)	1.45	192, (0.23)	1.45	129, (0.16)	1.50	103, (0.13)
	1.43	400, (0.60)	1.46	208, (0.30)	1.51	144, (0.29)	1.49	114, (0.17)
s298	0.02	2, (0.006)	0.02	2, (0.006)	0.02	2, (0.006)	0.02	2, (0.006)
	0.02	6, (0.013)	0.02	4, (0.009)	0.02	3, (0.008)	0.02	3, (0.009)
s420	7.25	1,170, (2.50)	7.67	669, (1.49)	9.16	556, (1.25)	8.99	423, (0.93)
	9.04	1,937, (4.28)	9.42	1,033, (2.30)	9.74	736, (1.64)	10.6	620, (1.42)
s444	0.69	42, (0.10)	0.69	39, (0.09)	0.69	33, (0.08)	0.69	33, (0.09)
	0.71	122, (0.29)	0.70	77, (0.18)	0.71	65, (0.15)	0.71	58, (0.14)
s641	0.80	37, (0.33)	0.78	27, (0.27)	0.74	25, (0.23)	0.77	26, (0.24)
	0.80	37, (0.32)	0.78	27, (0.24)	0.74	25, (0.23)	0.77	26, (0.24)
s713	0.49	24, (0.21)	0.48	17, (0.15)	0.44	16, (0.15)	0.44	16, (0.14)
	0.49	24, (0.22)	0.48	17, (0.16)	0.44	16, (0.15)	0.44	16, (0.15)
s838	36.5	3,856, (16.0)	48.2	2,743, (11.5)	61.6	2,154, (8.69)	109	2,322, (8.70)
	49.1	8,313, (45.2)	50.7	4,382, (18.5)	53.9	3,240, (13.9)	60.3	2,750, (12.5)
s1423	39.7	1,187, (100)	39.3	692, (60.3)	36.9	495, (42.1)	40.2	421, (36.6)
	45.7	1,931, (161)	46.1	1,027, (82.6)	43.4	683, (53.5)	41.3	540, (43.8)
s5378	13.6	596, (156)	14.4	376, (98.4)	15.4	300, (79.0)	15.5	279, (74.7)
	13.6	697, (173)	14.4	399, (103)	15.9	288, (71.1)	16.4	313, (77.0)
s9234	0.19	9, (1.96)	0.19	5, (1.19)	0.19	5, (1.14)	0.19	5, (1.12)
	0.19	9, (2.17)	0.19	5, (1.20)	0.19	5, (1.20)	0.19	5, (1.19)
s13207	313	1,564, (380)	342	865, (212)	362	601, (147)	373	682, (161)
	313	1,564, (360)	342	865, (210)	362	601, (150)	373	682, (161)
s38417	296	5,754, (35,481)	328	3,583, (23,237)	355	2,798, (18,374)	425	2,542, (15,941)
	295	5,813, (39,904)	333	3,659, (23,305)	352	2,695, (16,819)	426	2,566, (16,461)