C並行処理プログラムのテスト充分性評価システムのソケットへの対応

7M－3

川口豊＊ 伊東栄典＊ 古川善吾＊＊ 牛島和夫＊

＊ 九州大学工学部情報工学科, ＊＊ 九州大学情報処理教育センター

1. はじめに

並行処理プログラムの普及にともない、並行処理プログラムのテストが重要な特性となってきている。我々は、これまでにプロセス間通信・同期の実行順序を考慮したテスト基準を提案し、セマフォを用いたCプログラムに対して、そのテスト基準に基づくテスト充分性評価システムを試作した[2]。

今回、我々はプロセス間通信機構の一つであるソケットを用いたCプログラムにシステムを対応させた。ソケットはプロセス間通信方法として広く用いられている機構である。ソケットを用いたプログラムにシステムを対応させることにより、システムの適用範囲が広がる。本稿では、テスト充分性評価システムをソケットに対応させるに当たっての問題点について議論する。

2. 定義

2.1 順序列テスト基準

並行処理プログラムのテストでは、通信・同期を行なう文(**通信・同期命令**)の実行順序を考慮する必要がある。そこで、並行処理プログラムのテスト基準として、通信・同期命令を実行順に並べた列(**順序列**)をテストの測定対象とする**順序列テスト基準**を提案してきた[1]。

[定義 1] 順序列テスト基準

長さ n の順序列テスト基準

$$OSC_n = \{< s_1, \ldots, s_n >\}$$

s_i: 通信・同期命令、$< \ldots >$: 順序列、$\{\cdots\}$: 集合.

長さ2の順序列を特に**順序対**と呼び、長さ2の順序列テスト基準のことを**順序対テスト基準**という。順序対テスト基準では、任意の入力データに対して通信誤りを生じる「完全通信同期誤り」を検出することができる。

2.2 被覆率

テストの充分性を評価するに当たって、テスト充分性の指標となる値が必要となる。テスト充分性の指標値として、プログラム中の全測定対象に対する実行された測定対象の比(**被覆率**)を用いる。

3. 評価システム

我々が試作したC並行処理プログラムのテスト充分性評価システム[2]は、Cで記述されたソースコードを入力として受けとり、順序対テスト基準に基づく被覆率を出力する。

本システムはソースコードから通信・同期命令を抽出する。そしてプログラムの実行時に実行された通信・同期命令を検出する。

本システムは、(1) ソースコードを解析して、通信・同期命令を抽出する**ソースコード変換部**、(2) プログラムを実行させ、実際に実行された通信・同期命令を検出する**実行部**、(3) ソースコード変換部、実行部で出力されたデータから順序対テスト基準に基づく被覆率を計算する**被覆率計算部**で構成される。

3.1 モニタ

実行部では、本システムはソースコード変換部で変換されたソースコードをコンパイル、実行して、実際に実行された通信・同期命令を検出、記録する。システムは、通信命令の検出、記録をおこなうために別プロセス(**モニタ**)を使用する。

モニタの動作の手順は次の通りである。

i) 通信・同期命令の実行許可を受けつける。
ii) 通信・同期命令を特定する値や、実行検出に必要な値を受けとり、通信・同期命令が実行されたことを記録する。
iii) 通信・同期命令の実行許可を出す。
iv) 通信・同期命令が実行されたことを確認する。

4. ソケットへの対応

4.1 ソケット

ソケットは、プロセス間で双方向通信を行なう機構である。ソケットを介した通信には、プロセス間に通信路を確立して、データの到着を保証する**ストリーム型**と、通信路を確立せず、データの到着を保証しない**データグラム型**とがある。

4.2 実行検出

本システムは、ソケットによる通信を行なう通信命令の前後に、モニタに通信命令の実行開始と終了とを通知する探針を挿入する。

```
pre(...);          探針 (前: 実行開始)
read(..);
post(...);         探針 (後: 実行終了の通知)
```

図 1: 探針の挿入

ソケットからデータを読む通信命令は、データがない場合データが到着するまで実行を中断することがある。モニタが実行検出を探針からの通知にのみ頼っていると、通信命令が実行を中断した場合デッドロックを起こす可能性がある。そこで、モニタはカーネルメモリからプロセスの状態を調べ、プロセスがデータ待ち状態になっていた場合には通信命令が実行されたとみなして次の処理を行なう。

5. 例題

UNIX で実時間対話に用いられている phone プログラムに本システムを適用した。phone プログラムは、通信にソケットを用いている。プログラムのソースコードから抽出した通信命令数および測定対象としての順序対数、実際に phone を実行して得られた、全通信命令数、通信命令数、実行された順序対数、計算によって得られた被覆率、を表1に示す。

表 1 に示されているように、被覆率が 7.3 % と低い値である。通信・同期命令の実行記録を調べると、phone プログラムの中のエラー通知のために行なわれる通信・同期命令がほとんど実行されていないことが判った。すなわち、phone プログラムのデーモンプロ

Testing reliability evaluation system for C concurrent programs using sockets
Y.Kawaguchi, E.Itou, Z.Furukawa, K.Ushijima
Kyushu University.

表 1: phone に評価システムを適用した結果

プログラム解析結果	通信命令数	40
	順序対数	1600
実行結果	延べ通信命令数	461
	通信命令数	23
	順序対数	117
計測結果	順序対被覆率	7.3%

セスが、ファイルを開くのに失敗したといったエラーが発生したことをクライアントプロセスに知らせるための命令 7 個の中で実行されたのは、1 つだけである。これは、テストを意識して phone プログラムを実行した結果にではなく、普通に phone プログラムを使用した時の実行結果に基づいているためである。このことは、エラー通知だけでなく、通信命令の被覆率が約 60 % (23 命令 /40 命令) であることからも判る。この通信命令の被覆率は、並行処理動作に限定した時のテストにおける被覆率を示している。

しかしながら、被覆率が低い値になっている理由として、本システムが制御フロー上実行不可能な順序対を測定対象として取り込んでいることがある。次の節で実行不可能な測定対象について議論する。

5.1 実行不可能な測定対象

本システムは、プログラムの制御フローを考慮しないためにプログラムの動作上実行不可能な順序対を測定対象として取り込むことがある。例えば、図 2のプログラムにおいて、(このソースコードから生成されるプロセスが 1 つのみである場合には) 文 1 を実行した後、文 3 を実行する前に必ず文 2 が実行される。したがって、< 文 1, 文 3 > の順序対が実行されることはない。しかし、本システムはこの順序対を測定対象として取り込む。

```
sendto(id, ..); /* 文 1 */
...;
sendto(id, ..); /* 文 2 */
sendto(id, ..); /* 文 3 */
   /* <文 1, 文 3> の順序対は実行されない */
```

図 2: 実行不可能となる順序対の例

プログラムの制御フローを解析するプログラムとの協調や、テスト実施者の指示を受けて、評価システムが制御フローの上で実行不可能な順序対を減らすことによって被覆率をより正確にすることが必要となる [3]。

5.2 実行時間遅延の影響

通信を行なうプログラムには、データの到着を一定時間待機するという処理を行なうものがある。ソケットを使用したプログラムには、この処理を行なうものが多く見られる。そのようなプログラムを本システムに適用した場合、モニタによる実行遅延のため時間切れの処理しか行なわれなくなる可能性がある。

本システムは、通信・同期命令の実行順序を正しく記録するために通信・同期命令の実行を遅らせる。したがって、一定時間待機する処理を行なうところでは本システムによる動作の影響が無視できなくなる。そこで、通信・同期命令の実行時間が影響する場合について考察する。

データが到着するのを一定時間待機する方法として、以下の 2 つの方法がある。

- システムコール select を用いる方法
  システムコール select は、指定した入力からデータが読みだし可能になるまで待機する際に用いられるシステムコールである。このシステムコールは、待機時間を値に持つ変数を引数に与えることによって、待機時間を指定することができる。時間切れでシステムコールを終了したことは、システムコールの戻り値によって知ることができる。
  システムコール select を用いている場合には、本システムは select を測定対象として取り込んでおき、実行遅延が生じる可能性がある箇所としてテスト実施者に報告する必要がある。今回の場合、時間切れによる select の終了しか行なわれていないことも本システムによって実行記録から知ることができる。
- シグナルを用いる方法
  シグナルとは一種の割り込みで、プロセスはシグナルを受けとると、そのシグナルに対応した処理を行なう (シグナルを無視する場合もある)。シグナルを用いて一定時間データ到着を待つには、データを読む通信命令を実行する前に、待機時間後にシグナルを発生させるよう指定しておき、さらに時間切れの処理を行なう関数をシグナル受取時の処理として設定しておけば良い。
  現在、本システムはシグナルのような割り込み型のプロセス間通信方法に対応していない。シグナルを用いた時間待ち処理への対応は今後の研究課題である。

6. まとめ

我々は、プロセス間通信・同期を行なう文の実行順序を考慮した、並行処理プログラムのテスト基準に基づく C プログラム用のテスト充分性評価システムをソケットに対応させた。さらに本システムが実用的なプログラムに対して機能することを示した。本システムは、プログラムの制御フローにおいて実行不可能な事象を測定対象として取り込むことがある。

今後の課題として、(1) 他のプログラムに本システムを適用することによって、より多くの実行結果を検討すること、(2) テストケース生成系などとの連携を行ない、テストの支援を図っていくこと、が考えられる。

参考文献

[1] 伊東栄典、川口豊、古川善吾、牛島和夫:C 並行処理プログラムのプロセス間通信に関するテスト充分性評価について、情報処理学会研究会報告 Vol.93, No.13, 93-SE-90, pp.9–16, 1993.

[2] 川口豊、伊東栄典、古川善吾、牛島和夫:C 並行処理プログラムのテスト充分性評価システムについて、情報処理学会研究会報告 Vol.94, No.6, 94-SE-96, pp.107–114, 1994.

[3] 伊東栄典、川口豊、古川善吾、牛島和夫: 順序列テスト基準における測定事象の最適化方法について、第 49 回情報処理学会全国大会論文集 (掲載予定), 1994.