ビジネスコンピュータのための性能予測ツール

1P−7

松澤 智子, 小川 直樹

三菱電機(株) 情報電子研究所

1 はじめに

ビジネス分野ではユーザシステムの規模は拡大するにつれて、システム設計時に必要なSEの技術力は増大している。特に大規模なシステムの設計においては、SEの経験と技術に依存するところが大きく、性能の予測のためには

- 待ち行列理論など専門的な知識が必要
- 予測のための手順が標準化されていない

などの問題点があり、有効な手段が確立されていない。

これらの課題を解決し、システムの基本設計の段階で、システムの処理能力、端末応答時間を効率よく予測するために、本ツール（以下、SMARTとする）を試作した。

2 SMARTの特長

SMARTの特長は以下の通りである。

1. 簡単な定義で業務のモデル化が可能である
   - COBOLや当社簡易言語（PII）等の事務処理用言語の入出力命令、ソート命令、ライブラリ命令に即して各業務の処理手順を記述することが可能。
   - 待ち行列理論など性能評価の専門知識を必要としないので、解析的な評価よりも容易に性能予測を行なえる。
2. モデルシステムの動作がアニメーションで確認できる
   - シミュレーション実行期間中、モデルシステムの動作をアニメーション画面により確認できる。（後出図2参照）
   - シミュレーション実行途中でシステム構成に関するパラメータを変更し、モデルシステムの状態変化も即座に知ることが可能。

3 SMARTを用いた評価手順

SMARTを用いて評価を行なう時の手順について示す。

1. 机上でのシステム設計
   システム構成、トランザクション量、業務の処理手順などを予め仮定する。
2. 定義ファイルの作成
   1での設計結果をモデル定義言語DEFを用いて表記する。業務のモデル定義は、大まかにモデル定義、リソース定義、アプリケーション定義の3つの部分からなる。
3. モデル操作
   モデルの登録と、必要に応じてモデルの複写／削除を行なう。
4. トランザクション量の登録／変更
   1時間単位、最大24時間分のトランザクションの発生量を設定する。
5. シミュレーション実行
   シミュレーション中のシステム動作状態が、アニメーションにより表示される。
6. 結果の出力
   業務のスループットやリソースの負荷、端末応答時間に関する8種類のグラフを表示する。

4 SMARTでのモデリング

4.1 モデル定義部分

これは、利用者が最初に定義する部分で、システム特性を設定する。定義項目は、機種モデル名、主メモリサイズ、OSメモリサイズなどである。

4.2 リソース定義部分

これは、システム構成のうちトランザクションの待ちが発生する要素（リソース、例えば、CPU、ディスク等）について定義する部分である。表1にモデル定義部分での定義項目を示す。

表1: リソース定義部分での定義項目

タイプ	内容
CPU	通常は利用者の指定なしに、暗黙的なリソースとして扱われる
ディスク装置	平均ディスクアクセス時間、サービス規則(到着順等)を指定する
レコードロック ファイルのロック	データアクセスのパターンは一様と仮定 データ数を仮定する
アプリケーションの多重度	多重度、サービス規則を指定する
その他	ユーザシステム独自のリソース属性を記述する

4.3 アプリケーション定義部分

これは、アプリケーション定義文（処理形態等）とアプリケーション処理命令文（内部処理）を指定する部分である。

アプリケーション定義文では、アプリケーションの種類をトランザクション処理、ＴＳＳ処理、バッチ処理等から選び、その属性としてトランザクション等の投入間隔やジョブのプライオリティ等を設定する。

また、SMARTで予め用意したアプリケーション処理命令文の他に、ユーザがリソースの処理を記述することも可能である。

表2にアプリケーション処理命令文について示す。

Simulation Measurement Analysis and Reporting Tool: SMART
Tomoko Matsuzawa, Naoki Ogawa
Mitsubishi Electric Corp.

表 2: アプリケーション処理命令文

分類	数	内容	
フロー制御	7	ループ (LOOP) 分岐 (GOTO) ラベルまで無条件／確率的に分岐 呼びだし (CALL) サブルーチンの呼びだし。RETURN で復帰 リンク (LINK) 他のアプリケーションを起動 スリープ (SLEEP) 実行を遅延させる 終了 (END) リソースの解放とアプリケーションフローの終了	
アプリケーション処理	29	ファイルアクセス	例 ISAM.READ.RANDOM(リソース名): 索引ファイルのリード DAM.WRITE(リソース名): 相対ファイルのライト
		ファイルオープン	例 ISAM.OPEN(): 索引ファイルのオープン
		ファイルクローズ	例 DAM.CLOSE(): 相対ファイルのクローズ
		端末入出力	例 RGET()、RPUT()
		ソート	例 SORT()
		マージ	例 MERGE()

フロー制御での ( ) 内や、アプリケーション処理での「例」は、DEF 言語の表記法である。

5 評価例

あるバッチ業務を例に挙げ、SMART によってユーザの要求する性能条件を満たすシステム構成を決定する手順を説明する。

5.1 定義ファイル

定義ファイルの一部として、アプリケーション JOB 1 のフロー部分を以下に挙げる。

```
 1  APPLICATION   JOB1
 2      TYPE   BP;
 3      DISTRIBUTION   RANDOM;
 4      COBOL-LINE   100;
 5  {
 6    ALOC   RG;
 7    LOOP   9
 8          ISAM.READ.RANDOM(DISK1);
 9    ENDLOOP
10    LOOP   6
11          ISAM.REWRITE.RANDOM(DISK2);
12    ENDLOOP
13    LOOP   7
14          ISAM.WRITE.RANDOM(DISK3);
15    ENDLOOP
16    END;
17  }
```

図 1: 定義ファイルの一部

1～4 行目では、JOB1 がバッチ処理タイプのアプリケーションであることを示し、その属性としてプログラムは COBOL100 行から成り、ジョブの投入間隔がランダムであることを表している。8、11、14 行目では各々索引ファイルに対する read、rewrite、write のオーバヘッドをシミュレートしている。また、6 行目ではトランザクションが 1 リージョンを専有し、16 行目では全てのリソースを解放してアプリケーションが終了することを表している。

5.2 シミュレーション結果の例

図 2に示した画面からシミュレーション実行中、会話的にパラメータの変更ができるという SMART の機能を用いて評価を行なった。

まず、ある機種モデルにおいてシミュレーションを行なった。この場合、応答時間が大きくなり、CPU がほぼ常時使用され、他のリソースが空き状態になった。つまり、この機種モデルでは、CPU ネックが生じることがわかった。

そこで、実行中の画面で CPU 性能を上げた設定に変更し、引続きシミュレーションを行なった。この場合には、CPU や他のリソースがバランスよく使用され、応答時間が短くなり、スループットも向上することがわかった。

従って、このような業務を処理する場合には、機種モデルを上げる必要のあることが、シミュレーションにより明らかになった。

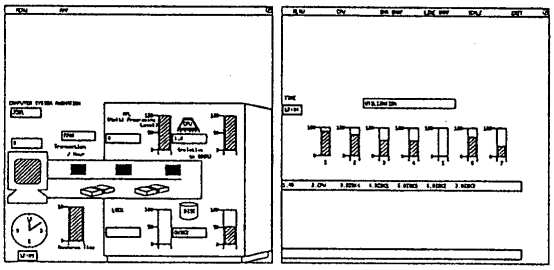

図 2: シミュレーション実行中の画面の例

6 終わりに—今後の課題—

SMART を用いることにより、システム設計時の作業効率の向上がみれらた。しかし、未だ完全なツールではなく、以下のような課題が残っている。

1. 複雑な業務をモデル化するのが困難である
2. システムの動的条件（ユーザ数増加に伴うカーネルオーバヘッドの増大等）を考慮していないため、予測結果の誤差が大きくなる場合がある

更に SMART を使い易く、精度の高いものにするため、これらの課題を解決しなくてはならない。

参考文献

[1] 松澤、小川他： OLTP 性能予測ツール SMART
情報処理学会 第 4 2 回全国大会 講演論文集