オペレーティング・システム　30－6
（１９８６．２．２１）

待ち行列モデルによる計算機システムの性能評価エキスパートシステムの設計

村瀬 勉　　西田 竹志　　宮原 秀夫　　高島 堅助

大阪大学 基礎工学部 情報工学科

内容梗概

待ち行列モデルを用いて,計算機システムの性能評価を行うエキスパートシステムの設計について述べる．このシステムは待ち行列理論の知識と対象計算機の知識からなるモデル構築知識ベースと性能改善策知識ベース,及びモデル解析プログラムである解析部で構成されている.本システムはモデル化をする際に必要なリソースモニタデータを,モデル構築知識ベースを用いて推論し,使用者に問い合わせることによって,実システムに対応する待ち行列モデルを自動的に構築する.使用者は,対応するデータをリソースモニタデータから取り出し入力するだけで,待ち行列に関する知識を持たなくても,容易に実システムに対応する待ち行列モデルの構築が行える.また,モデル解析部が性能評価を行った後,性能改善策知識ベースの知識を用いることによって,計算機の利用状況に合った性能改善案を提示することができる．ここでは,それぞれの知識ベースの内容について,モデルと実システムとの対応付けを主にして,例を挙げて説明する.

1.はじめに

計算機システム(以下単にシステム又は実システムと呼ぶ)の設置,再設定を行うに当たっては,現在のシステムの利用状況,使用効率などのデータから,そのシステムが適切に効率良く運用されているかどうかを知り現状のシステムでの問題点を明確にした上で,それに見合ったシステムの導入がなされるべきである.

しかしながら,計算機システムの性能評価の現状として,計算機システムのオペレータが,システムの利用効率などを知りたいと思うとき,リソースのモニタリングを行っているにも関わらず,その評価,例えば<1>リソースモニタから得られる情報に対する評価<2>システムの改善案による性能の変化予測,については計算機システムの専門家（システムエンジニア等）のノウハウや経験的手法,試行錯誤等を用いてなされていることが多く系統的な性能予測が行われていないと思われる.

計算機システムの系統的な性能評価を行う手法として,待ち行列モデルによる解析,あるいはシミュレーションによる方法,等が知られているが,これらの方法は待ち行列理論の専門家と計算機システムの専門家の共同作業を必要とし,これが実際に性能評価をする上で,大きな制約になっている.つまり(1)実システムを表現する待ち行列網モデルの構築,(2)実システムの利用状況データをどのような形でモデルの入力として用いるか(3)待ち行列モデルを解析し性能評価測度を算出する(4)性能評価の結果に対してシステムを有効に使用するには具体的にどこを改善すればよいのかの４点を行うことが待ち行列網理論に精通していないオペレータにとっては容易ではない.

ところが,これまでに開発された性能評価パッケージにおいては,待ち行列モデル(以下単にモデルと呼ぶ)の解析パッケージが多く,主に待ち行列の専門家によって使用される傾向にあり,(1),(2),(4)のステップを含んでいるものではなかった.なぜなら,(1),(2),(4)の部分では実システムや待ち行列理論の知識と知識を扱う推論機構を必要とするからである． そこで,(3)を中心に(1),(2),(4)の機能を知識ベースと推論機構という形で備えたエキスパートシス

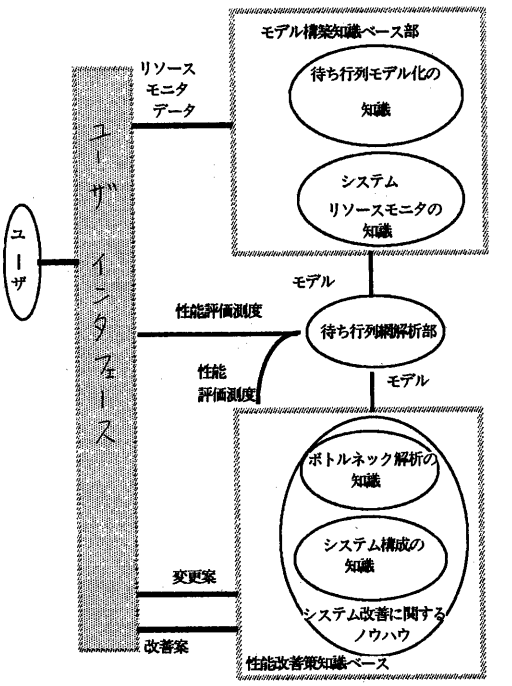

図1　エキスパートシステムの構成図

テムが必要となる．

このエキスパートシステムは(1),(2),(4)の機能を実行するための待ち行列モデル構築知識ベース，性能改善策知識ベースと，待ち行列網解析プログラムであるモデル解析部の３つの部分と推論部から成る(図１)．このエキスパートシステムの目的は，実システムから得られるリソースモニタデータのデータを，エキスパートシステムと会話しながら入力することによって，システムの性能評価を行い，さらに性能改善策知識ベースの提供する改善案によりモデルを変更し，この改善案に従ってシステムを変更した際の性能を予測することにある．

これにより計算機オペレータは待ち行列理論の知識なしに，システムの性能評価ができ，さらにシステムの性能予測ができる．

２　エキスパートシステム

２．１　エキスパートシステムの概要

計算機システムの性能評価の一般的手順を分析すると次のようになる．(図２参照)

なお以下ではジョブのレスポンスタイム（ＴＳＳユーザが会話型ジョブをシステムに投入してからジョブが完了しシステムが次のプロンプトを端末に送ってくる迄の時間）をシステムの性能の主な評価測度とする．

①（システム動作の記述★）
ジョブを待ち行列におけるトランザクションとみなし，このジョブがどのリソースをどれくらい使用するかを，ＯＳの動作（特にスケジューラ，スワッパー，ディスパッチャー）をふまえて解析する．

②（システムリソースの待ち行列モデル化★）
ジョブの使用するリソースを，レスポンスタイムに影響するレベルで待ち行列に置き換えジョブが使用するリソースの順序から，システムに対する待ち行列網モデルを構成する．

③（待ち行列モデルのパラメータの設定★）
②の待ち行列モデルのパラメータ値を実システムのデータから算出する．ここで実システムのデータとは，実システムのリソースモニタデータであり，モデルパラメータとは，例えば "CPUを表す待ち行列へのTSSジョブの到着率" である．

④（待ち行列モデルを解析して評価測度を計算）
待ち行列モデルを解析して各待ち行列のサーバの利用率，平均待ち時間等とレスポンスタイムを算出する．

⑤a（実システムの変更案★）
性能改善の為の実システムのＯＳのパラメータやシステム構成の変更案の作成．

⑥a（変更案のモデルへの反映★）
⑤a の変更案がモデルではどこに影響するかを考え，モデルを変更する．

⑤b（待ち行列モデルにおける改善案《ボトルネック解析等》★）
性能改善のため待ち行列モデル上で改善案を作成し，そ

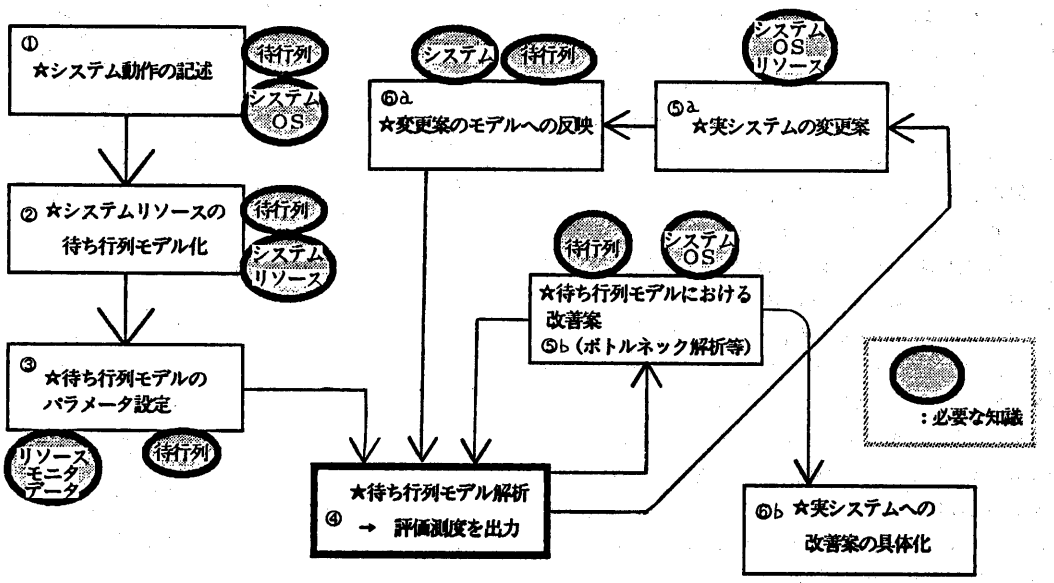

図2　性能評価手順　と　必要な知識

れに従ってモデルを変更する.

⑥b (実システムへの改善案の具体化☆)

⑤b のモデル変更は実システムではどのように具体化されるのかを示す.

つまり,モデルの作成(①②③)の後は,モデルの解析(④)とモデル(実システム)の変更(⑤),実システム(モデル)への反映(⑥)のステップを繰り返すことになる.

このうち④のステップは待ち行列モデル解析ツールとしてパッケージ化されている.しかしながらその他の部分,特に☆の印を付けたステップでは,待ち行列に関する知識とその計算機システムの動作に関する知識を同時に要求される.つまりシステム動作の記述からシステムのモデルを作成し,リソースモニタデータからモデルのパラメータを設定するにはシステムのＯＳに関して,且つ待ち行列理論に関しての知識が必要である. 又システムを評価した後システムの性能向上を計る為の改善案を示し性能を予測する際にも,同様の知識が必要である.

そこでこれらのステップを待ち行列専門家の知識と実システムに関する知識を持つエキスパートシステムとして構成することにより,計算機システムのオペレータがエキスパートシステムと対話しながらシステムのチューンナップを行うことができる.

図１に,このエキスパートシステムの構造概念図を示す. また,その特徴を次に挙げる.

<<<< 特徴 >>>>

☆各リソースをその処理方式を考えて適当な待ち行列の時間分布やサービス規律を決定する.

(例　CPU　→　指数時間サービス ,
Processor Sharing(PS) サービス規律)

☆モデル解析部はシミュレーションではなく,ＢＣＭＰ型待ち行列網を数値解析で解いているので,比較的短い時間で計算できる.

☆ユーザは,尋ねられたデータを端末から入力することによって,会話的に性能評価測度を得られる.

☆モデル構築の際データが不足する場合,待ち行列に関する知識と推論機構により入力データのほかに,どんなデータセットがあれば,モデルを解析するのに十分なデータがそろうかを推論し,これを提示する (後述) .

☆知識ベースの知識を用いて実システムでの改善案をモデルの変更に変換する.

☆モデル解析の結果からの性能改善案を作成し,また実システムにおける改善案を示す.

２.２　待ち行列モデル構築知識ベース

モデルについて

システム (システムリソース) を待ち行列網としてモデル化して評価するときに決定すべきものとして,どのリソースを待ち行列としてモデル化するか,各待ち行列の接続形態 (ルーティング) と,各待ち行列での処理(サービス)時間分布,到着時間分布,サービス規律 (待ち行列パラメータ) がある.

基本的な待ち行列構成要素とＯＳの動作を考慮すると,ほとんどのシステムがChandyが[1]で述べたモデルにあてはまる.

モデル(図３)はChandyのモデルにバッチジョブのオープンチェインを加えたもので,スケジューリングやスワッピングの遅延についてはＣＰＵのオーバヘッドとして解析している.

図３は一般的なＴＳＳターミナル,主記憶,ＣＰＵ,DISK(I/Oデバイス) のクローズドチェイン (ＴＳＳジョブチェイン) と,ソース(バッチジョブの入力),シンク(同出力),主記憶,ＣＰＵ,DISKのオープンチェイン (バッチジョブチ

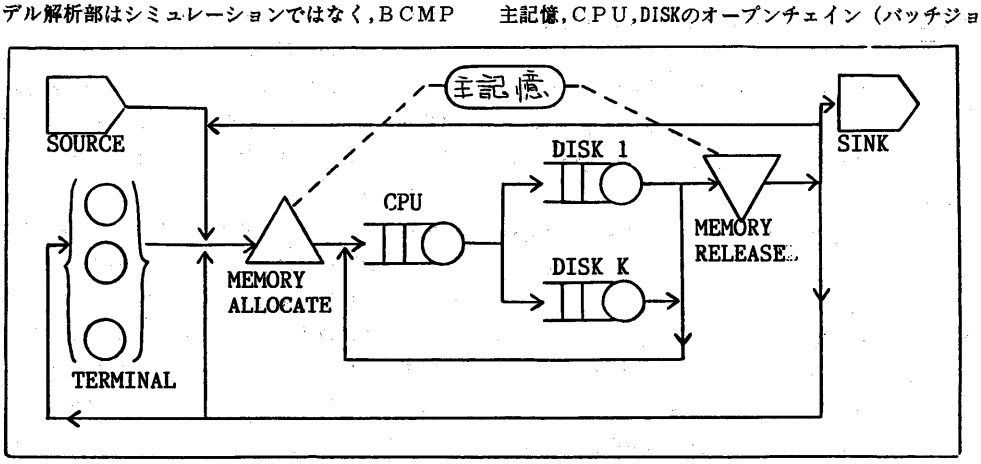

図３　待行列網モデル

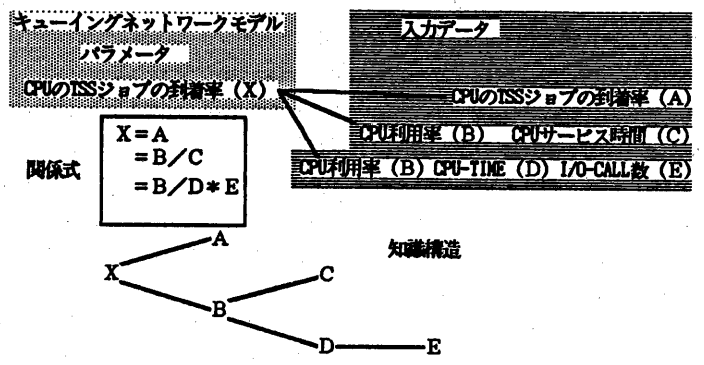

図4 パラメータ と 入力データ の関係

ェイン）の2チェインモデルを表している．

各待ち行列のサービス時間分布の関数の型(指数 一定)などはデバイスの性質によって決定されるとする．従ってパラメータとして，サーバ数，平均サービス時間，到着率を実システムに応じて決定する必要がある．

実システムのデータとモデルパラメータの関係

ここでは，実システムのデータからのモデルのパラメータの決定方法について述べる．

一般にリソースモニタのデータは直接にモデルの入力パラメータにはならない。従って，リソースモニタデータをモデルに合うように加工する必要があるが，これには，システムの動作，ＯＳの動作と待ち行列との対応付けが必要である．

そこで，リソースモニタデータからモデルのパラメータを算出すること，システム変更によるモデルの変化について考えなければならない．

ここでは入力データとパラメータの関係について例を挙げて説明する． リソースモニタで直接測定できないモデルパラメータは以下の例のように別のパラメータを用いて算出できる(図４)．

これは，ＣＰＵを表す待ち行列へのＴＳＳジョブの到着率(X)は入力データ(リソースモニタデータ)の複数のデータセットから算出できることを表している．

このような式を知識として持つことによりＣＰＵのＴＳＳジョブの到着率というモデルパラメータが算出できる．

又，システムのリソースモニタデータではそのシステム固有の言葉の定義や概念があり，一般に計算機システムごとに異なっている．従ってそれ等を待ち行列理論で使用する言葉との定義の変換を行う知識も必要となる．

例

＜某システム＞

レスポンスタイム=ＴＳＳジョブが起動されてから最初のＣＰＵサービスを受けるまでの時間

ターンアラウンドタイム=応答時間

＜待ち行列理論＞

レスポンスタイム=ＴＳＳジョブが端末から起動されてから次のプロンプトが端末へ帰ってくるまでの時間（応答時間）

ターンアラウンドタイム=バッチジョブが起動されてから結果が出力されるまで．

従ってこのシステムでは

システムのターンアラウンドタイム=モデルのレスポンスタイム

と解釈する必要がある．

本エキスパートシステムはこれらの知識を持つ知識ベースをもとにモデルに必要なデータに対応するリソースモニタデータを推論し使用者に問い合わせるので待ち行列に関する知識を持たない人でも，システムの待ち行列モデルが構築できる．

２．３ 性能改善策知識ベース

待ち行列モデルの解析（モデル解析部）でシステムの性能評価測度が算出できたならば，その数値についての分析を行わなければならない．そのとき，システムが負荷が増加した時にレスポンスが急に悪化するといったことがあるか，将来そのようなことが起こるかどうか，又その時どのようにシステムを改善するべきかについて考えなければならない．それらの問題に対していろいろな改善法がある．それは大きく２つにわけると，システムの動作解析の結果や計算機システムの専門家が経験的に知っている改善方法のノウハウ等と，待ち行列網の動作解析より導かれるモデル上での改善方法である．

前者では実システムの変更をモデルの変更に置き換える必要から，後者ではモデルの変更が実システムでは具体的にどのような形の変更になって現れるのかを考えるのに，実システムの知識，待ち行列の知識が必要になる．ところがこれまでの性能評価パッケージでは，性能評価後の対策はシステムオペレータ等が，実システムの改善案を作った後，変更点をモデルに入力しなければならなかった，あるいは，待ち行列専門家がモデル上において改善案を作成していた．

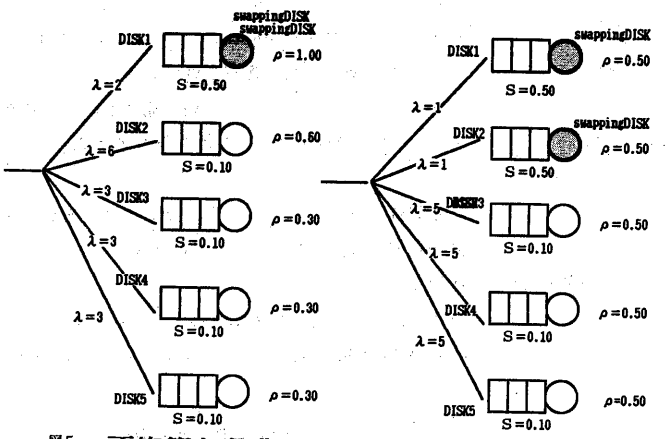

図5　不均等な負荷 →→→ 負荷の均等化

DISK1がボトルネックの原因　　ボトルネックの解消

そこで実システムの知識,待ち行列の知識からなる知識ベースによって改善案におけるモデルと実システムの整合をとることで改善案を採用したときの性能予測をすることができる.

モデル上での改善案を作成する方法の一例としてボトルネック解析の結果を用いる方法がある. この方法は,システムの動作を漸近的に解析するもので性能改善の方法が明確な形で与えられている.

ボトルネック解析は,システムに高負荷がかかったとき,利用率が１の待ち行列(ボトルネックノード)のスループットがシステムのスループットを決定してしまうという状態(飽和状態)が起こり,レスポンスタイム等が飛躍的に悪化するという現象を解析するものである.

ここではボトルネック解析の結果を知識としてどのように利用するかについてのべる.

ボトルネック状態を解消する,つまり利用率を下げる方法としては待ち行列理論により待ち行列モデル上において次の２つの策が考えられる

1 サービス時間Ｓを短縮する

2 到着率λを減らす

1,2 については

1 サーバのキャパシティを大きくする

2a 分岐確率を変える

2b 新たなノードを追加する

2c バッシブキュー[1]の容量変更

等の方策が考えられる

モデル上でのこれらの方策は実システムではそれぞれ

1.デバイスの性能を上げ処理スピードをアップする

2a.処理の代行を行える機器で利用率の低いものがあればそこへ処理要求を廻す

2b.同じ処理を行うデバイスを追加設置する

2c.ＯＳのマルチプログラミングレベルを変更する

等の対策を採ることで実現できる.

ところがモデル上では考えられないが,あるパラメータの変更が全く別のいくつかのパラメータの変化を引き起こすといったことがありうる.例えば2aの例でいえばスワッピングをほとんど行わないようにシステムパラメータを変更する,つまりswapping専用DISKへの到着率を下げるとその影響は平均ＣＰＵタイムの増加,CPU待ち行列のサービス時間の増加,swapping専用DISK 以外のDISKへの到着率の減少として現れる.従ってこのことを知識として知識ベースに登録しておいてswapping専用DISKの到着率の変更の際にはこの知識でモデルの他のパラメータも変えなければならない. しかしＴＳＳジョブが存在するシステムでこれらの変更を実行するためには,さらに,主記憶の容量を増やすかマルチプログラミングレベルを小さくするかしなければならない.なぜならばスワッピングを行わないことは主記憶割り当てにおいてＦＣＦＳつまりバッチ処理の形態を示すことになりＴＳＳ処理ができなくなるからである.よって,このことも知識として登録しておく必要がある.

例として,５台のDISKのうちDISK1がボトルネック状態を引き起こしているシステムに対し改善案2a,2bを用いてボトルネック状態を解消した結果を図５に示す.

３．プロトタイプシステム

知識ベースのプロトタイプをVAX11/VMS上でのプロダクションシステムＯＰＳ５でインプリメントした.

プロトタイプとして, リソースモニタデータから待ち行列モデルのパラメータを算出するための知識（ルール）をＯＰＳ５でインプリメントし,モデル解析部で評価測度(レスポンスタイム,利用率 等)を計算した後,グラフ出力するシステムを作った.

システム構成

＜待ち行列モデル構築知識ベース部＞

○データ入力と入力データの前処理（ＯＰＳ５）

・ユーザは,評価する計算機のリソースモニタデータを入力することを要求される.

・モデルパラメータを求めるのに必要なデータセットを

ユーザに訪ねるための知識.

○モデルのパラメタの算出（ＯＰＳ５）

・リソースモニタデータの数値からモデルパラメータを算出する知識.

○不足しているデータの検出,再入力 （ＯＰＳ５）

・あるパラメータを求めるのに,不足しているデータセットを知らせる知識.

＜モデル解析部＞

○２チェイン　ＢＣＭＰ型クローズドネットワークの解法

○Flow-equivalent 法 (aggregation 法)

＜出力インターフェース＞

○評価測度の出力（Ｃ）

・response time　VS. TSSユーザ

等のグラフ出力.

４。実行結果

プロトタイプの実行結果の一例として,モデルパラメータの設定のための会話(リソースモニタデータの入力)後の,エキスパートシステムの応答(不足データの検出)(図6)とその解析出力（レスポンスタイム VS. TSSユーザ数）を,図７に示す.

５。あとがき

計算機システムの系統だった性能評価をするためには,モデル化を行う知識,システムの改善案を作成する知識,及びそれらを推論する機構を持つ性能評価エキスパートシステムが必要である． ここで述べたシステムは,待ち行列モデル解析プログラムを中心にして,モデル構築知識ベース,性能改善策知識ベースで構成されている． このシステムと対話することによって待ち行列理論についての知識がない人でも,計算機システムの性能評価ができる.

謝辞

本研究を行うにあたって大阪大学基礎工学部の下条真司氏から多大な助言をいただき,感謝いたします.

参考文献

1)Charles H.Sauer,K.Mani Chandy "COMPUTER SYSTEMS PERFORMANCE MODELING", chap. 9.4 PRENTICE-HALL 1981

エキスパートシステムとの会話

エキスパートシステムの質問		ユーザの応答
★ＴＳＳジョブの平均利用者数	: -->	
★端末での平均思考時間	: -->	
★ＴＳＳジョブの平均レスポンスタイム	: -->	
★平均会話時間	: -->	
★平均総セッション時間	: -->	
★平均総会話数	: -->	
★ディスク(I/Oデバイス)台数	: -->	12
★ＣＰＵ個数	: -->	1
★ＴＳＳジョブのスワップイン回数／単位時間	: -->	
★バッチジョブのスワップイン回数／単位時間	: -->	0.016667
★ＴＳＳジョブの平均総ＣＰＵ時間	: -->	0.0136
★ＴＳＳジョブの平均総Ｉ／Ｏコール回数	: -->	522
★ＴＳＳジョブのＣＰＵ利用率	: -->	

（以下　略）

エキスパートシステムは,以下のデータが不足していることを知らせる.

ＴＳＳジョブのＣＰＵへの到着率　　を得るためには
** ＴＳＳジョブのＣＰＵ利用率
　の データセットが必要です

端末の平均サービス時間　　を得るためには
** 平均思考時間
　の データ　　または
** 平均総セッション時間
** 平均総会話数
** TSS-ジョブの平均レスポンスタイム
　の データセット　または
** 平均会話時間
** TSS-ジョブの平均レスポンスタイム
　の データセットが必要です

TSS-ジョブの分岐確率Ｂ　　を得るためには
** TSS-ジョブのスワップイン回数／単位時間（秒）
** TSS-ジョブの平均利用者数
** 平均会話時間
　の データセットが必要です

図６　　エキスパートシステムとの会話

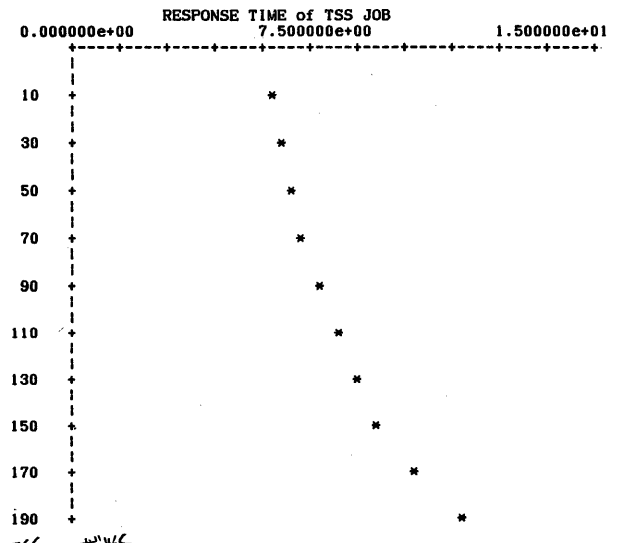

図７　Response Time　vs. TSSユーザ数