双安定特性を有する計算機・通信システムの性能評価に関する研究

Studies
on
Performance Evaluation
of
Computer and Communication Systems
with Bistable Characteristics

横平徳美

1989年2月

内容梗概

一般的に，計算機および通信システムなどにおいて，それらのシステムの状態を表わす変数（状態変数）に関して，対象とするシステムが安定して動作する状態値を1つしか持たない時，そのシステムは単安定モードにあるという．また，システムが2つの安定状態値を持つ時，そのシステムは双安定モードにあるという．トラヒック等のシステムの状態に影響を及ぼすパラメータ（以下，コントロールパラメータと呼ぶ）の値の変化に対し，単安定モードにも，双安定モードにもなり得るシステムを双安定特性を有するシステムという．このような双安定特性を有するシステムは，コントロールパラメータの変化から引き起こされる安定モードの変化により，システム性能が急激に悪化する可能性があると言われている．よって，コントロールパラメータの変動に対して，双安定特性を有するシステムを効率よく動作させるためには，(1)システムが双安定モード，単安定モードになるコントロールパラメータの領域を求め，(2) 安定モードとシステム性能がどのように関連しているか，といったことを考察する必要がある．

そこで本論文では，双安定特性を有する以下のシステムに対して，それらシステムのコントロールパラメータ，安定モードおよび性能の関連性について，カタストロフ理論，特にくさびのカタストロフ（ *Cusp Catastrophe* ）を用いて考察する．

- 通信路（チャネル）へのアクセス方式として，ランダムアクセス方式を用いた通信システム，
- マルチプログラミング方式を用いた計算機システム（マルチプログラミングシステム），
- 回線交換ネットワークにおいて，ルーティング方式として *alternate-routing* を用いた通信システム．

まず，ランダムアクセス方式を用いた通信システムにおいて，パケットの到着率，再送率をコントロールパラメータとし，システムが双安定モードになる領域(*Cusp*)を導出する．さらに，コントロールパラメータが *Cusp* を横切るように変化した時に，システム性能（平均バックログ，スループット，平均パケット伝送遅延）が急激に変化するということを明らかにするとともに，トラヒックの増加に対して，システム性能の急激な変化を防ぐ方法について述べる．

次に，マルチプログラミングシステムに対し，ジョブの発生率，マルチプログラミングレベルの最大値をコントロールパラメータとして，くさびのカタストロフを応用

することにより，システム性能（スループット，レスポンスタイム）と *Cusp* の関係を明らかにする．さらに，任意のジョブの発生率に対してスループットを最大にし，かつ，レスポンスタイムを最小とするようなマルチプログラミングレベルの最大値が存在することを示す．

最後に，回線交換ネットワークにおける５つのルーティング方式，つまり *non-alternate-routing (NAR), alternate-routing (AR), alternate-routing-trunk-reservation (AR-TR), alternate-routing-external-blocking (AR-EB)* および *alternate-routing-trunk-reservation-and-external-blocking (AR-TR-EB)* に対して，呼損率（*end-to-end-blocking-probability : EEBP*）をシステム状態として，くさびのカタストロフを応用し，それぞれの方式の安定性を比較・検討する．さらに，ネットワークの負荷の増加に対して，*Cusp* を用いてルーティング方式を動的に切り換える方式として２つの方式を提案し，これら２つの方式の有効性を明らかにする．

関連発表論文

A. 論文誌関係

[1] T. Yokohira, T. Nishida and H. Miyahara, "Analysis of Dynamic Behavior in p-persistent CSMA/CD using Cusp Catastrophe," *COMPUTER NETWORKS and ISDN SYSTEMS*, Vol. 12, No. 5, pp.277-289, 1986.

[2] 横平徳美, 宮原秀夫, "マルチプログラミングシステムにおけるカタストロフィー現象の解析," 電子情報通信学会論文誌(D), Vol.J71-D, No.6, pp. 966-973, 1988.

[3] 横平徳美, 宮原秀夫, "回線交換ネットワークにおける種々のルーティング方式の安定性の比較," 電子情報通信学会論文誌(B), Vol.J71-B, No.12, pp. 1411-1418, 1988.

B. 専門研究会

[4] 横平徳美, 西田竹志, 宮原秀夫, 高島堅助, "カタストロフィー理論を用いた p-persistent CSMA/CD 方式の性能評価," 電子情報通信学会技術研究報告, CS85-59, (1985,8).

[5] 横平徳美, 宮原秀夫, "マルチプログラミングシステムにおけるカタストロフィー現象の解析," 電子情報通信学会技術研究報告, SE87-48, (1987,7).

[6] 横平徳美, 宮原秀夫, "回線交換ネットワークにおける種々のルーティング方式の安定性の比較," 電子情報通信学会, 情報ネットワーク・交換ワークショップ, (1988,3).

目 次

1. 序論

多重アクセスチャネルを用いたローカル・エリア・ネットワーク，衛星通信などの通信システムにおいて，チャネルに対する有効なアクセス方式として，*ALOHA* 方式[1,2]，*CSMA/CD(Carrier Sense Multiple Access with Collision Detection)*方式[3,4]のようないわゆるランダムアクセス方式がある．

これらのランダムアクセス方式は基本的に，各ノードが他のノードとは独立にパケットを送信する方式であり，次のような利点がある．

- パケットを送信する際の手続きが簡単である．
- 伝送制御のためのオーバーヘッドが小さい．
- ノードの追加，除去が簡単である．

しかしながらランダムアクセス方式では，パケット送信の際にチャネル上で他のノードから送信されたパケットとの衝突が起こる可能性があり，衝突を起こした各ノードは，あらかじめ定められた確率分布に従う時間間隔後に，この衝突したパケットを再送しなければならない（この再送時間間隔の平均の逆数を再送率という）．

ランダムアクセス方式を用いたシステムにおいて，システムのトラヒックが低い，つまり各ノードへのパケットの到着率が小さいときには，パケット同士の衝突はあまり起こらないので，パケットの再送回数は比較的少ない．よって，低トラヒックの時には，定性的に，システム内に滞留するパケットの個数が少ないような状況でシステムが動作する．つまり，システム内に滞留するパケットの個数をシステムの状態を表わす状態変数とした場合，低トラヒックの時には状態変数の小さな値でシステムは安定に動作する．この状態を非飽和安定状態という．また，システムが唯一の安定状態しか持たず，かつ，その状態が非飽和安定状態であるとき，そのシステムは非飽和安定モードにあるという．

逆に，システムのトラヒックが高い時には，パケットの衝突の可能性が増し，パケットの送信が成功するまで，何回もそのパケットを再送しなければならないという状況に陥る．よって，高トラヒックの時には，システム内に滞留するパケットの個数は多くなり，システムは状態変数の大きな値で安定に動作するようになる．（この安定状態を飽和安定状態と呼び，唯一の飽和安定状態で稼働しているとき，システムは飽和単安定モードにあるという）．

また，パケットの到着率，再送率の値によっては，システムは非飽和安定状態と飽

和安定状態という2つの安定状態を持ち，パケット到着の確率的変動により，2つの安定状態の間で発振(*thrashing*)するような状況が生まれる場合がある．システムがこのような状況にあることを双安定モードにあるという．

このようにランダムアクセス方式は，パケットの到着率，再送率によって，システムの動作するモードは非飽和単安定モード，飽和単安定モードおよび双安定モードの3通りになる．また，双安定モードになり得る可能性を有するシステムは双安定特性を有するシステムと呼ばれる．

いま述べた双安定特性は，マルチプログラミング方式を用いた計算機システム（マルチプログラミングシステム）においても存在する[5,6]．

各ユーザーから発生される各々のジョブは，主記憶（以下，メモリと呼ぶ）を分割使用しながら処理される．マルチプログラミングシステムにおいて，ある時点で，メモリを分割使用しているジョブの数をマルチプログラミングレベル(*MultiProgramming Level* : *MPL*)という．

マルチプログラミングシステムにおいて，*MPL* が小さい，つまり各ユーザのジョブの発生率が小さい（低トラヒック）時には，*page-fault* はあまり発生せず，システムは効率的にジョブを処理するので，システム内に滞留するジョブの個数は非常に少ない（非飽和単安定モード）．しかし，ジョブの発生率が大きい（高トラヒック）時は，*page-falut* が多発し，システム内に滞留するジョブは非常に多い（飽和単安定モード）．よって，マルチプログラミングシステムにおいては，*page-fault* の過度の多発を防ぐためには，*MPL* に対して，最大値(*Maximum MultiProgramming Level*: *MMPL*)を設定する必要があるが，ジョブの発生率，*MMPL* の値によっては，ランダムアクセス方式と同様に，非飽和安定状態と飽和安定状態が同時に存在し，ジョブの発生率の確率的変動により，2つの安定状態の間での発振が起こる可能性がある（双安定モード）．

これら双安定特性を持つシステムに対するこれまでの研究において，*ALOHA* 方式に対しては，[1,7,8,9,10,11,12,13] において，各安定モードの関係についての検討が行われており，特に，[8,9,10,11]において，パケットの到着率が大きくなるにつれて，再送率が比較的小さい時には，システムの安定モードは，双安定モードを経験することなしに，非飽和単安定モードから飽和単安定モードに変化し，システム性能（スループット，平均パケット伝送遅延等）はゆるやかに変化するが，再送率が大きすぎる場合には，システムの安定モードは双安定モードを介して，非飽和単安定モードから飽和単安定モードに変化し，それにつれてシステム性能は急激に変化する，と述べられている．

このようにランダムアクセス方式を用いたシステム，マルチプログラミングシステムのように双安定特性を有するシステムにおいて，トラヒックの変動に対して，システムを効率的に動作させるためには，各安定モードの関連性について調べる必要があると思われる．いいかえればシステム設定時のパラメータであるパケットの再送率，*MMPL* がある値に設定されたとき，システムの入力パラメータであるパケットの到着率，ジョブの発生率の変化に対して，システムの安定モードがどのように変化するかということを知る必要がある．ランダムアクセス方式に対するこれまでの研究において，*ALOHA* 方式に対しては，いま述べたように各安定モードの関係が明らかにされている．しかしながら，*CSMA/CD* 方式，マルチプログラミングシステムに対しては，それらが双安定モードになる可能性があると述べられているにすぎず，各安定モードの関係はまだ十分に明らかにされていない．*CSMA/CD* 方式は *Ether-Net* 等の多くのローカル・エリア・ネットワークに，またマルチプログラミング方式は多くの計算機システムに用いられており，その安定モードの解析も重要と思われる．

文献[9,10,11]においては，*ALOHA* 方式の安定モードの変化を調べるにあたってカタストロフ理論[14,15]を用いている．カタストロフ理論はシステムの状態に影響を及ぼすパラメータ（以下，コントロールパラメータと呼ぶ）の変化に伴って，状態が急激に変化するようなシステムに対して応用することができる．カタストロフ理論において，この状態の急激な変化はカタストロフと呼ばれ，この理論を用いることによって，コントロールパラメータとシステム状態の関係を定量的に明らかにすることができる．よって，ランダムアクセス方式を用いたシステムにおいては，パケットの到着率，再送率を，また，マルチプログラミングシステムにおいては，ジョブの発生率，*MMPL* をそれぞれのコントロールパラメータとして，各々のシステムにカタストロフ理論を応用することにより，コントロールパラメータの変化と状態の変化（ひいては安定モードの変化）の関係を明らかにすることができる．

また，回線交換ネットワークにおけるルーティング方式のひとつである *alternate-routing (AR)* においても双安定特性は存在することが知られている[16,17,18,19]．

回線交換ネットワークにおいて，ルーティング方式は *non-alternate-routing (NAR)* と *AR* の２つに大別することができる．*NAR* がただ一つの決まった *path(fixed-path)* しか使用しないのに対し，*AR* は，*fixed-path* が使用できないときには，迂回路 *(alternate-path)* を用いるルーティング方式である．これら２つの方式における *end-to-end-blocking-probability (EEBP)* は，図 1.1 に示すように，ネットワークのトラヒックが小さい（ノードへの *call* の到着率が小さい）ときには *AR* の方が，また，トラヒックが大きいときには，*EEBP* は *NAR* の方がそれぞれ小さくなる．さらに，トラヒックの増加に対して，*NAR* における *EEBP* は比較的ゆるやかに変化するが，*AR* におい

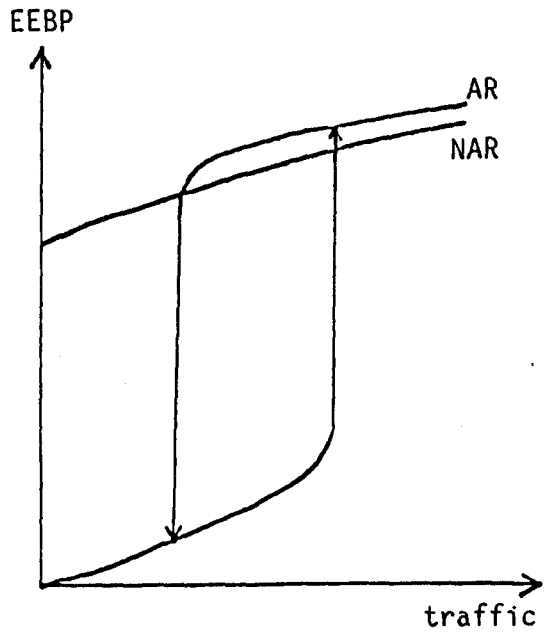


図 1.1: AR における $EEBP$ の双安定特性

ては，*EEBP* が急激に劣化する．また，図 1.1 からわかるように，*EEBP* をシステムの状態とした時，トラヒックのある値に対しては，*EEBP* が 2 つの値になっている．つまり，同じトラヒックの値に対して，*EEBP* が小さい安定状態（非飽和安定状態）と *EEBP* の大きい安定状態（飽和安定状態）が存在する．よって，*AR* が双安定モードになることがわかる．

AR における *EEBP* の急激な劣化を防ぐ一つの方法として，*alternate-routing-trunk-reservation (AR-TR)* 方式が提唱されている[18]．そこでは *fixed-path* に対して，あらかじめ少数のリンクを予約することにより *EEBP* の急激な劣化を防ぐことが示されているが，予約するリンクの数が不適当な時には，*AR* と同様に双安定モードになり，トラヒックの変動に対して *EEBP* は急激に変化する．

また，[16] において，*alternate-path* にかかるトラヒックをある確率で却下するルーティング方式 *(alternate-routing-external-blocking: AR-EB)*，また，*AR-TR* と *AR-EB* を結合して用いるルーティング方式 *(alternate-routing-trunk-reservation-and-external-blocking: AR-TR-EB)* が提唱され，*NAR*，*AR*，*AR-TR*，*AR-EB* および *AR-TR-EB* の 5 つのルーティング方式の性能比較がなされてはいるが，それぞれのルーティング方式において，トラヒックがどのような値の時に双安定モードになるのか，また，トラヒックがどのような値の時に *EEBP* の急激な変化が起こるか，といったいわゆる安定性の議論は行われていない．ルーティング方式の安定性はネットワークの性能と深く関わっており，各々の方式の安定性について知ることは，トラヒック変動に伴うネットワークの性能を評価する上で重要なことと思われる．

本論文では，いままで述べてきたような双安定特性を有する計算機・通信システムに対して，カタストロフ理論を用いて性能評価を行う．つまり，システムのトラヒックの変動に対して，システムの安定モードがどのように変化するか，また，それに伴いシステム性能がどのように変化するか，ということについて議論する．

本論文は次のような構成になっている．第 2 章ではカタストロフ理論，特にくさびのカタストロフについて述べる．システム状態の急激な変化が，カタストロフ理論で定義されるポテンシャル関数の形状変化と関係していることを明らかにする．第 3 章ではくさびのカタストロフをランダムアクセス方式に応用する．まず，[9]において，くさびのカタストロフを *ALOHA* 方式に応用する際に用いられた方法について述べる．次に，[9] の方法を用いて，くさびのカタストロフを *p-persistent CSMA/CD* 方式に応用し，システムの安定モードがパケットの到着率，再送率の変化に対して，どのように変化するか，ということについて議論する．特に，システムが双安定モードになるコントロールパラメータの領域 *(Cusp)* を導出し，トラヒックの変化に対する

システム性能（スループット，パケット伝送遅延）の変化が *Cusp* と深く係わっていることを明らかにする．第4章ではくさびのカタストロフをマルチプログラミングシステムに応用し，第3章と同様の議論を行う．第5章では回線交換ネットワークにおけるルーティング方式の安定性をくさびのカタストロフを用いて比較，検討する．各ルーティング方式の *EEBP* を導出し，トラヒックと *EEBP* の関係がカタストロフ理論におけるカタストロフィー多様体に類似することを示し，*Cusp* を導出する．そして，*Cusp* を用いて各ルーティン方式の安定性を議論する．

2. カタストロフ理論

ここでは，カタストロフ理論[14,15]，特にくさびのカタストロフの概要を述べる．システム状態の急激な変化が，ポテンシャル関数の臨界点に関する形状変化から起こるということを明らかにする．

2.1 ポテンシャル関数

実軸上の閉区間$[u,v]$からなる状態空間上において，あるひとつのシステムを考え，そのシステムの状態をスカラー連続値$x(\in[u,v])$，システムのコントロールパラメータをベクトル$\mathbf{c}=(c_1,c_2,\ldots)$とする．実数値関数$P(x;\mathbf{c})$において，状態$x$と，その近傍の状態$x'$を考えた時，任意の$\mathbf{c}$に対して，$P(x;\mathbf{c})\geq P(x';\mathbf{c})$である時に，システムの状態が$x$から$x'$に遷移する時，$P(x;\mathbf{c})$をポテンシャル関数と呼ぶ．これは，任意に設定されたコントロールパラメータにおいて，ポテンシャル関数の極小点に対応する状態にシステムが移動し，システムがその状態で安定するということを意味する．

カタストロフ理論の主な定理であるトムの分類定理(*Thom's classification theorem*)[14,15]によって，2個のコントロールパラメータをもつポテンシャル関数は，コントロールパラメータをa,bとすると次の多項式関数で与えられる．

$$P(x;a,b)=x^4+bx^2-ax \tag{2.1}$$

次に，ポテンシャル関数$P(x;a,b)$に関連して，その臨界点の型，およびドリフト関数について定義する．

定義1：臨界点の型

与えられたa,bに対して，$P'(x;a,b)=0$となる点，つまり臨界点(*critical point*)は$P''(x;a,b)$の符号により次のような3つの型(*type*)の点に分類される．

- 負臨界点　：$P'(x;a,b)=0$かつ$P''(x;a,b)<0$
- 正臨界点　：$P'(x;a,b)=0$かつ$P''(x;a,b)>0$
- 退化臨界点：$P'(x;a,b)=0$かつ$P''(x;a,b)=0$

定義2：ドリフト関数

$P(x;a,b)$ に対して，$-P(x;a,b)$ の微分$-P'(x;a,b)$ をドリフト関数 $d(x;a,b)$ と呼ぶ．つまり，

$$d(x;a,b) = -4x^3 - 2bx + a \tag{2.2}$$

定義からわかるように，$d(x;a,b)$ の値が正の時システムは状態 x が増加する方向に移動し，負の時には状態 x が減少する方向に移動することを意味する．

ポテンシャル関数 $P(x;a,b)$ の典型的な形状とそれに対応するドリフト関数 $d(x;a,b)$ を図 2.1 に示す．$P(x;a,b)$ の曲線上にのっているボールをシステムとみなし，システムの状態をボールの水平方向の値とする．図 2.1 においてA点，C点は正臨界点，B点は負臨界点である．また，ドリフト関数曲線において x 軸上の矢印は，$d(x;a,b)$ の定義よりわかるようにシステムの状態が移動する方向を表わす．A点，B点，C点に対応する状態 x_1，x_2，x_3 において $d(x;a,b)$ の値は 0 であるので，A点，B点およびC点のいずれかの点に対応する状態にシステムの状態がある時，システムはその状態にとどまる．よって，臨界点に対応する状態 x_1，x_2，x_3 は平衡状態(*equilibrium state*)と呼ばれる．システムの状態は，状態 x_1 の左側で正の方向に，右側で負の方向に移動するので，システムの状態が最初に状態 x_1 の近傍にあれば，ある時間後に状態 x_1 に移動して安定する．状態 x_3 についても同様である．よって，状態 x_1，x_3 は安定(*stable*)な平衡状態といわれ，これより正臨界点は安定平衡点といわれる．状態 x_2 においては，状態 x_1，x_3 とは逆に，状態 x_2 の左側で $d(x;a,b)$ の値は負，右側で正なので，たとえ，システムが一時的に状態 x_2 にあったとしても，外部からの小さな変動によってシステムはすぐに状態 x_1または x_3 に移動しようとする．よって，状態 x_2 は不安定(*unstable*)な平衡状態といわれ，これより負臨界点は不安定平衡点と呼ばれる．また，退化臨界点に対応する状態は退化平衡状態といわれる．

以上，述べたことをまとめると，平衡状態の型(*type*)は次のように3通りに場合分けされる．

定義3：平衡状態の型

- 安定　：正臨界点（安定平衡点）に対応する状態の型
- 不安定：負臨界点（不安定平衡点）に対応する状態の型
- 退化　：退化臨界点に対応する状態の型

また，平衡状態に関して，平衡状態の順序(*sequence*)を次のように定義する．

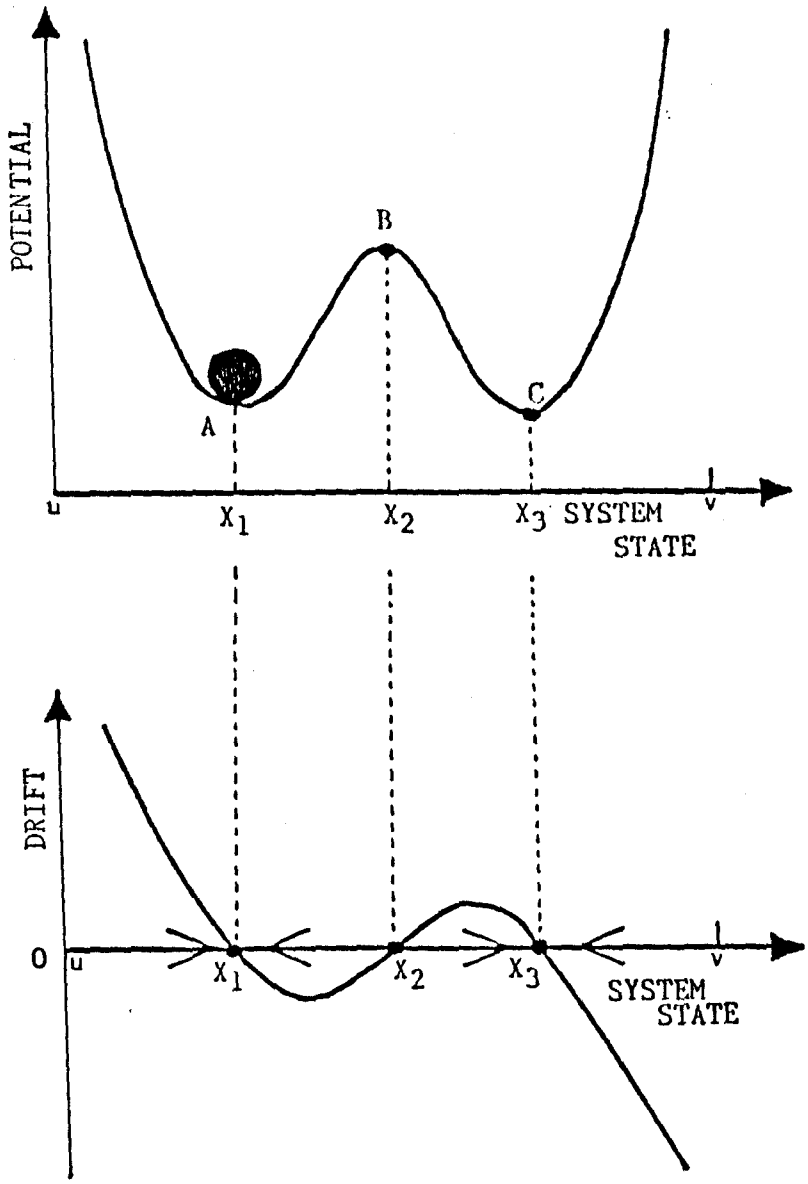


図 2.1: ポテンシャル関数 $P(x;a,b)$, ドリフト関数 $d(x;a,b)$

定義4：平衡状態の順序

ポテンシャル関数に関して，x 軸上を u から v へと見て入った時に出会う平衡状態の型の並びを平衡状態の順序という．例えば，図 2.1 において平衡状態の順序は（安定→不安定→安定）である．

2.2 くさびのカタストロフ

図 2.2a は，$d(x;a,b)=0$ を 3 次元空間(a,b,x) 上に表わしたものである．図 2.2a はカタストロフ理論において，カタストロフィー多様体と呼ばれている．ドリフト関数の定義からわかるように，カタストフィー多様体は，コントロールパラメータ(a,b) が与えられた時，システムが安定する状態の値を示している．また，カタストロフィー多様体の折り目を(a,b) 平面上に写像したものは，くさび$(Cusp)$と呼ばれる(図 2.2b)．$Cusp$ は，カタストロフィー多様体の折り目を(a,b) 平面上に写像したものであるから，$Cusp$ の式は，次の(2.4)式，(2.5)式の連立方程式を解いて得られる(2.5)式で与えられる．

$$P'(x;a,b)=0 \tag{2.3}$$

$$P''(x;a,b)=0 \tag{2.4}$$

$$8b^3+27a^2=0 \tag{2.5}$$

カタストフィー多様体をみればわかるように，点(a,b) が $Cusp$ の内部（図 2.2b の斜線部）にある時，ドリフト関数において，3 個の平衡状態が存在する，言い替えると，ポテンシャル関数は 3 個の平衡状態をもつ．また，点(a,b) が $Cusp$ の外部（図 2.2b の斜線部以外）にある時，ポテンシャル関数は 1 個の平衡状態をもつ．このように，$Cusp$ はポテンシャル関数の平衡状態の個数が変化する境界となっていることがわかる．さらに，ポテンシャル関数の平衡状態の個数だけではなく，平衡状態の順序も考慮した場合，ポテンシャル関数は $Cusp$ によって，次の 5 通りに場合分けされる．

(1) 点(a,b) が $Cusp$ の外側$(8b^3+27a^2>0)$ にある時，
平衡状態は 1 個，その順序は（安定）

(2) 点(a,b) が $Cusp$ の左側の曲線上$(8b^3+27a^2=0$ かつ $a<0)$ にある時，
平衡状態は 2 個，その順序は（安定→退化）

(3) 点(a,b) が $Cusp$ の頂点$(8b^3+27a^2=0$ かつ $a=0)$ にある時，
平衡状態は 1 個，その順序は（退化）

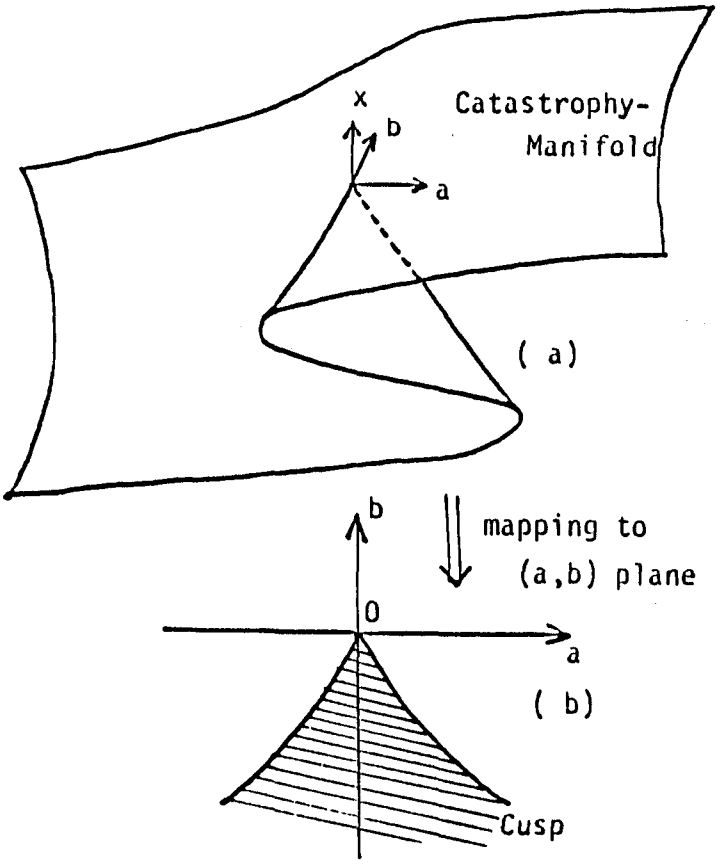


図 2.2: カタストロフィー多様体

(4) 点(a,b)が$Cusp$の右側の曲線上$(8b^3+27a^2=0$かつ$a>0)$にある時，
平衡状態は2個，その順序は（退化→安定）
(5) 点(a,b)が$Cusp$の内側$(8b^3+27a^2<0)$にある時，
平衡状態は3個，その順序は（安定→不安定→安定）

点(a,b)が$Cusp$の内部にある時，ポテンシャル関数は2個の安定平衡状態をもつような形状（双安定形状）になるので，システムは双安定モードにあるという．逆に，点(a,b)が$Cusp$の外部にある時，ポテンシャル関数は1個の安定平衡状態しか持たない形状（単安定形状）になるので，システムは単安定モードにあるという．

いま，bの値を固定して，aの値が変化した時の，状態xの変化について考える．図2.3aにbの値が正および負の時のカタストロフィー多様体の断面を示す．

図2.3aにおいて，$b<0$の時($trajectory$-1)，aの値が増加するにつれて，点(a,x)は1 → 2 → 3 → 4と変化し，2のところで，aの値の少しの増加に対して急激な変化（カタストロフ）Aが起こる．次に，このような状況で，aの値を減少させた時，点(a,x)は4 → 5 → 6 → 1と変化し，5のところで，aの値の少しの減少に対して急激な変化Bが起こる．

いま述べたことをポテンシャル関数の形状変化と対応させて述べる．図2.4に$P(x;a,b)$の形状の一例を示す．図2.4a，図2.4eは(1)の場合の形状，図2.4bは(2)の場合の形状，図2.4cは(5)の場合の形状，図2.4dは(4)の場合の形状の例である．図2.3aにおいて，aの値が増加するということは，図2.4において，$P(x;a,b)$の形状が図2.4aから図2.4eに変化することを意味する．$P(x;a,b)$の形状の図2.4aから図2.4dへと変化するにつれて，システムの状態はx_aからx_dへと変化するが，x_a，x_b，x_cおよびx_dはほとんど同じ値であり，状態はゆるやかに変化していることがわかる（この変化は図2.3aにおける1から2の変化に対応している）．ところが，図2.4dから図2.4eの変化においては，$P(x;a,b)$の形状がゆるやかに変化しているにもかかわらず，状態はx_dからx_eに急激に変化する（この変化は図2.3aにおける2から3の変化に対応している）．また，$P(x;a,b)$の形状が，いま述べた変化と逆の変化を起こした場合，図2.4bから図2.4aに形状はゆるやかに変化しているにもかかわらず，状態は急激に変化する（この変化は図2.3aにおける5から6の変化に対応している）．

次に，$b>0$の時には，図2.3aから明らかなように，aの値がどの様に変化しようと急激な変化は起こらない($trajectory$-2)．

いままで述べてきたことからわかるように，くさびのカタストロフでは，コント

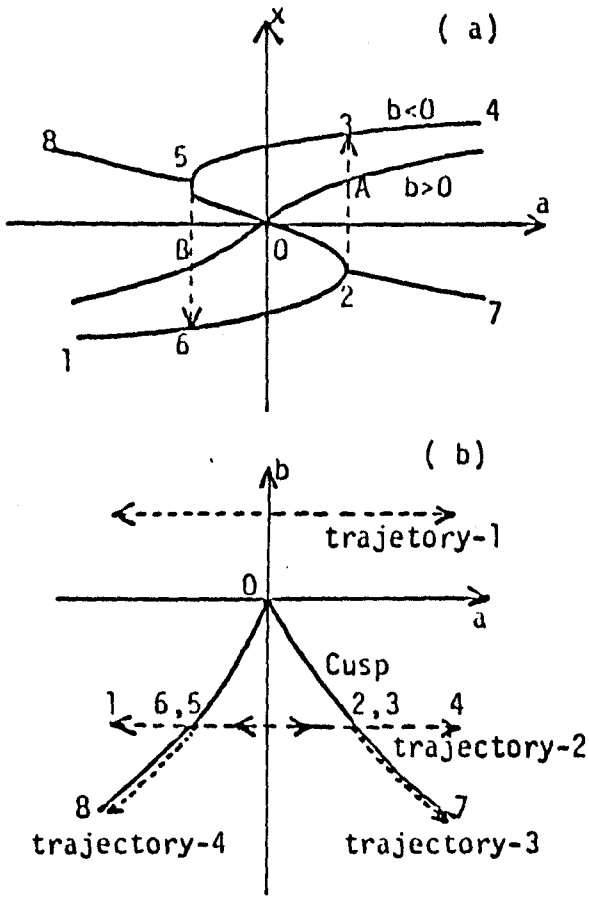


図 2.3: カタストロフィー多様体の断面

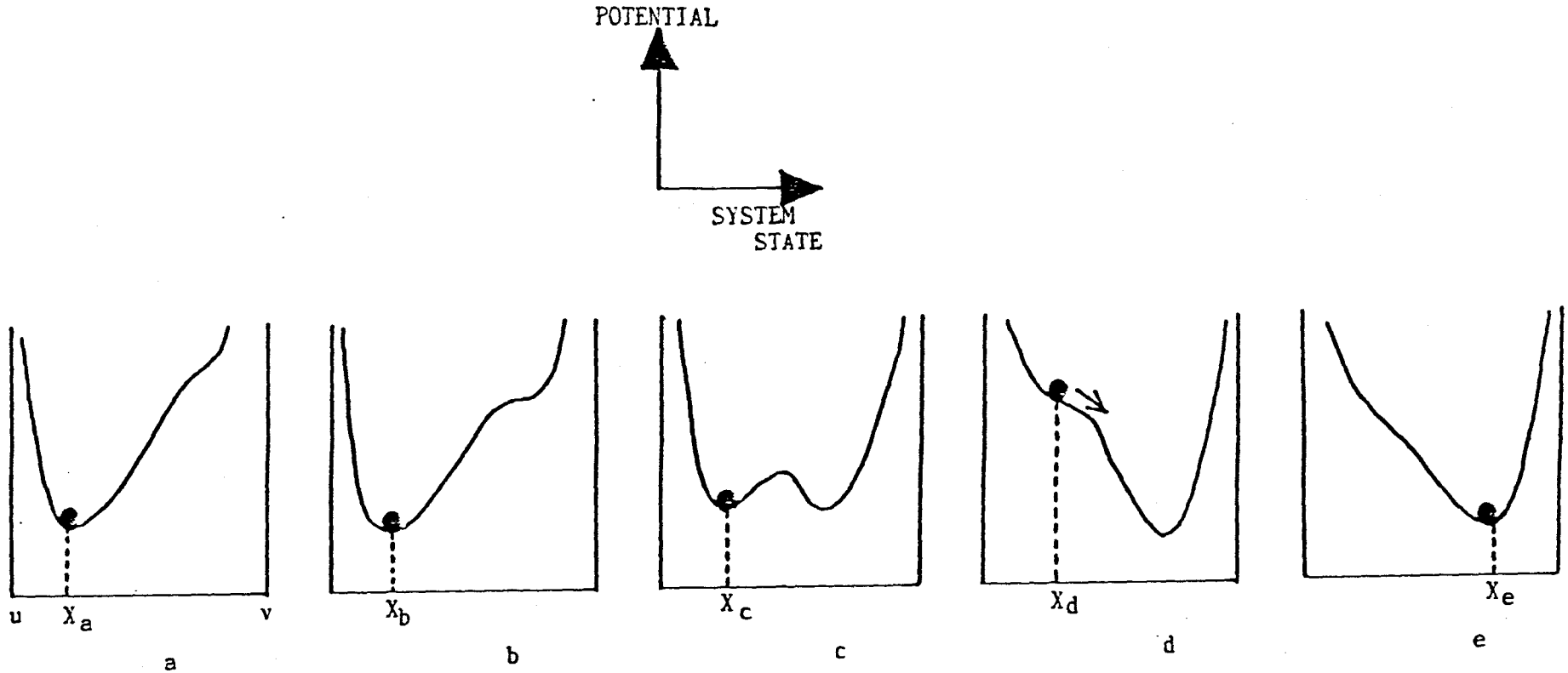


図 2.4: ポテンシャル関数の形状変化

ロールパラメータが *Cusp* を完全に横断したときに，急激な状態変化が起こり，また，状態変化においてはヒステリシス特性があることがわかる．また，a の値が変化したとき，b の値によって，急激な変化が起こるかどうかが決まる．よって，b はカタストロフ理論において，*splitting-parameter* と呼ばれる．また，a は *normal-parameter* と呼ばれる．

次に，a の値が増加した時に，b の値を *Cusp* の右側の曲線に従って減少させたとき（図 2.3a の *trajectory*-3），点(a,x) は 1 → 2 → 7 とゆるやかに変化し，また，a の値が減少してきた時に，b の値を *Cusp* の左側の曲線に従って減少させたとき（図 2.3a の *trajectory*-4），点(a,x) は 4 → 5 → 8 とゆるやかに変化する．このように，a の値の変化に対して，b の値を *Cusp* を用いて制御することにより状態の急激な変化を防ぐことができる．

3. ランダムアクセス方式の性能評価

[9,10,11,12,13]において，くさびのカタストロフを応用して，*slotted-ALOHA* 方式の安定モードとシステム性能の関係が解析されている．本研究では，くさびのカタストロフを *p-persistent CSMA/CD* 方式に応用する．まず，確率的なシステムのポテンシャル関数[9,10,11]を定義し，具体的に，このポテンシャル関数を用いてくさびのカタストロフを *slotted-ALOHA* 方式に応用した例[9,10,11]について簡単に述べる．そして，*p-persistent CSMA/CD* 方式に対して，くさびのカタストロフを応用し，システムの安定モードとその性能について，具体的に検討する[20,21].

3.1 確率的なシステムのポテンシャル関数

くさびのカタストロフを，後述するランダムアクセス方式，4 章で述べるマルチプログラミングシステムに応用する際には，それぞれのシステムの無限小平均，無限小分散[22]を導出し，それらを用いて，ポテンシャル関数を求める．ここでは，一般的な確率的システムの無限小平均，無限小分散について述べ，確率的システムのポテンシャル関数を定義する．

まず，図 3.1 のような *GI/G/1* 待ち行列を考える．図 3.1 における $A(t)$, $D(t)$ および $X(t)$ を次のように定義する．

$A(t)$: 時間間隔$[0,t]$ における客の待ち行列への総到着数
$D(t)$: 時間間隔$[0,t]$ における客の待ち行列からの総退去数
$X(t)$: 時刻 t における待ち行列内（サービス中の客も含む）にいる総客数

$A(t)$, $D(t)$ および $X(t)$ に対して，次の関係が成立する．

$$X(t) = X(0) + A(t) - D(t) \tag{3.1}$$

$\Delta A(t)$, $\Delta D(t)$, $\Delta X(t)$ を時間間隔$[t, t+\Delta t]$ における増加量とすると，

$$\begin{aligned} \Delta X(t) &= X(t+\Delta t) - X(t) \\ &= (A(t+\Delta t) - A(t)) - (D(t+\Delta t) - D(t)) \\ &= \Delta A(t) - \Delta D(t) \end{aligned} \tag{3.2}$$

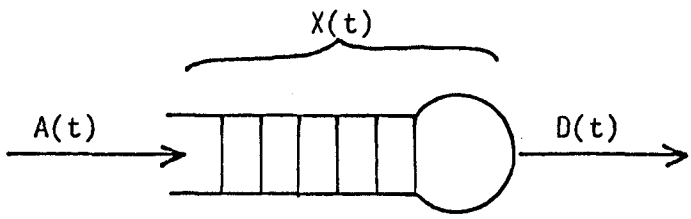


図 3.1: *GI/G/1* 待ち行列

よって，$\Delta X(t)$ の平均と分散は次のようになる．

$$E[\Delta X(t)] = E[\Delta A(t)] - E[\Delta D(t)] \tag{3.3}$$

$$V[\Delta X(t)] = V[\Delta A(t)] + V[\Delta D(t)] - Cov(\Delta A(t), \Delta D(t)) \tag{3.4}$$

ここでは，式(3.4)における $Cov(\Delta A(t), \Delta D(t))$ は無視できるほど小さいと仮定する．この仮定は待ち行列が高負荷のときは妥当のものとなる．つまり，待ち行列への到着が多いときには，$X(t)$ は大きくなり，$D(t)$ が $A(t)$ に依存する確率は極端に小さくなり，$A(t)$ と $D(t)$ がほとんど独立とみなすことができる．

ここで，$1/\mu_a(x,t)$，$1/\mu_d(x,t)$，$\sigma_a^2(x,t)$ および $\sigma_d^2(x,t)$ を次のように定義する．

$1/\mu_a(x,t)$： $X(t) = x$ のときの客の平均到着間隔
$1/\mu_d(x,t)$： $X(t) = x$ のときの客の平均退去間隔
$\sigma_a^2(x,t)$： $X(t) = x$ のときの客の到着間隔の分散
$\sigma_d^2(x,t)$： $X(t) = x$ のときの客の退去間隔の分散

$\Delta A(t)$,$\Delta D(t)$ は非負の整数値となるが，ここでは，中心極限定理より，$\Delta A(t)$，$\Delta D(t)$ をそれぞれ平均 $1/\mu_a(x,t)$，$1/\mu_d(x,t)$，分散 $\sigma_a^2(x,t)$，$\sigma_d^2(x,t)$ の正規分布で近似する[22]．

よって，$\Delta X(t)$ の平均と分散は次のようになる．

$$E[\Delta X(t)] = (\mu_a^3(x,t) - \mu_d^3(x,t))\Delta t \tag{3.5}$$

$$V[\Delta X(t)] = (\mu_a^3(x,t)\sigma_a^2(x,t) + \mu_d^3(x,t)\sigma_d^2(x,t))\Delta t \tag{3.6}$$

ここで，$X(t)$ の無限小平均$\mu(x,t)$，無限小分散$\sigma(x,t)$ を次のように定義する[22]．無限小平均，無限小分散は定性的に，無限に小さい時間における $X(t)$ の増加量の平均と分散に相当する．

$$\mu(x,t) = \lim_{\Delta t \to 0} \frac{E[\Delta X(t)]}{\Delta t} \tag{3.7}$$

$$\sigma(x,t) = \lim_{\Delta t \to 0} \frac{V[\Delta X(t)]}{\Delta t} \tag{3.8}$$

よって，無限小平均，無限小分散は次の式で与えられる．

$$\mu(x,t) = \mu_a(x,t) - \mu_d(x,t) \tag{3.9}$$

$$\sigma(x,t) = \mu_a^3(x,t)\sigma_a^2(x,t) + \mu_d^3(x,t)\sigma_d^2(x,t) \tag{3.10}$$

定常状態においては時間変動がないので，定常状態における無限小平均，無限小分散は次の式で与えられる．

$$\mu(x) = \mu_a(x) - \mu_d(x) \tag{3.11}$$

$$\sigma^2(x) = \mu_a^3(x)\sigma_a^2(x) + \mu_d^3(x)\sigma_d^2(x) \tag{3.12}$$

ここで，$1/\mu_a(x)$, $1/\mu_d(x)$, $\sigma_a^2(x)$ および$\sigma_d^2(x)$ は次のように定義されたものである．

$1/\mu_a(x)$: 定常状態 x における客の平均到着間隔
$1/\mu_d(x)$: 定常状態 x における客の平均退去間隔
$\sigma_a^2(x)$: 定常状態 x における客の到着間隔の分散
$\sigma_d^2(x)$: 定常状態 x における客の退去間隔の分散

次に，定常状態における無限小平均，無限小分散を用いて，確率的なシステムのドリフト関数について考える．

$T(y)$ をシステムが，最初に y という状態に到達するまでの時間とし，$f(\varepsilon)$ を次のように定義する．

$$f(\varepsilon) = Prob[T(x+\varepsilon) < T(x-\varepsilon) | X(t) = x] \tag{3.13}$$

$f(\varepsilon)$ は，時刻 t に状態 x にあったシステムが，状態 $x-\varepsilon$ に最初に到達する前に，状態 $x+\varepsilon$ に最初に到達する確率を表わす．$f(\varepsilon)$ は次の式で与えられる．

$$f(\varepsilon) = \frac{e^{2\mu\varepsilon/\sigma^2} - 1}{e^{2\mu\varepsilon/\sigma^2} - e^{-2\mu\varepsilon/\sigma^2}} \tag{3.14}$$

$\varepsilon \to 0$ を考えると，

$$f(\varepsilon) \cong \frac{1}{2}(1 + \frac{\mu\varepsilon}{\sigma^2}) \tag{3.15}$$

このように，システムの状態が微小な量$\varepsilon(>0)$ だけ移動する確率は，μ/σ^2 の関数になっていることがわかる．よって，確率的なシステムのドリフト関数を次のように定義する．

$$d(x) = \frac{2\mu(x)}{\sigma^2(x)} \tag{3.16}$$

次に，確率的なシステムのポテンシャル関数について考える．くさびのカタストロフにおいて，ドリフト関数はポテンシャル関数を微分し，-1 を乗じて得ることができる．よって，ドリフト関数からポテンシャル関数を得るためには，ドリフト関数を

積分して，-1 を乗じれば得られる[9,10,11]．よって，確率的なシステムのポテンシャル関数を次のように定義する．

$$V(x) = -\int_0^x d(x)dx \tag{3.17}$$

このように確率的なシステムのポテンシャル関数を得るためには，そのシステムの無限小平均，無限小分散を求めればよい．

3.2 slotted-ALOHA方式の性能評価

3.2.1 slotted-ALOHA 方式

ALOHA 方式は，衛星パケット通信網のようなノードが比較的広い地域に分散している環境において，提唱されているアクセス方式である．送信要求の起こったノード（送信ノード）は，まず，親ノード（衛星パケット通信網においては，衛星が親ノードに相当する）にパケットを送信し，親ノードによって，このパケットは宛先のノード（受信ノード）に送信される．送信ノードから親ノードへは，チャネルおよび他ノードの状況は全く考慮せずにパケットの送信が行われる．このとき他のノードが送信中であればチャネル上でパケットの衝突が起き，親ノードはパケットを正しく受信できない．親ノードはパケットを正しく受信した場合のみ確認パケット(*Ack* パケット)をそのパケットの送信ノードに送る．親ノードからの確認パケットが到着しない時には，送信ノードはパケットの衝突が起きたとみなし，ある一定時間間隔後に再送を試みる．

ALOHA 方式の代表的なものとして，各ノードが他のノードとは非同期にパケットを送信する *pure-ALOHA* 方式と，各ノードが送信のための共通のクロックを持ち，同期してパケットを送信する *slotted-ALOHA* 方式が挙げられる．*slotted-ALOHA* 方式は *pure-ALOHA* 方式と比較して，同期送信の効果によって，パケット衝突の確率が減少し，最大のスループットは 2 倍になることが知られている．

3.2.2 解析モデル

slotted-ALOHA 方式を解析するために，[9,10,11]において設けられた解析モデルの仮定を下記に示す．

- 送信するパケット長は固定とし，それを送信するために必要な時間をスロットと呼ぶ．
- 全ノード数を M とする．
- 各ノードはスロットに同期してパケットの送信を行なう．
- 各ノードは 1 パケット分のバッファをもっている．
- パケットを持っているノードはそれ以上のパケットは発生しない．

図 3.2 に *slotted-ALOHA* 方式の解析モデルを示す．

(1) 送信すべきパケットを持っていないノードは *Think* モードにあるという．*Think* モードにあるノードに対して，パケットの送信要求が確率 p_0 で発生する．
(2) チャネルに対してパケットの送信を行なう．
送信が成功した場合，*Think* モードになる．
送信が失敗した場合，*Backlog* モードに入る．
(3) *Backlog* モードにあるノードは各スロットの最初の時点で，確率 p でパケットの再送を試みる．

3.2.3 ポテンシャル関数

slotted-ALOHA 方式の状態を，その時点での *Backlog* モードにあるノードの合計とし，それを x で表わす．いま述べた解析モデルにおいて，定常状態における無限小平均 $\mu(x;p_0,p)$，無限小分散 $\sigma^2(x;p_0,p)$ は次のようになる[9,10,11]．

$$\mu(x;p_0,p) = (N-x)p_0 - (px + p_0(N-x))e^{-(px+p_0(N-x))} \tag{3.18}$$

$$\sigma^2(x;p_0,p) = (N-x)p_0 + (px + p_0(N-x))e^{-(px+p_0(N-x))} \tag{3.19}$$

よって，*slotted-ALOHA* 方式のポテンシャル関数 $V(x;p_0,p)$ は式(3.17)より次のようになる．

$$V(x;p_0,p) = -\int_0^x \frac{2\mu(x;p_0,p)}{\sigma^2(x;p_0,p)}dx \tag{3.20}$$

3.2.4 Cusp とシステム性能

[9,10,11]において，確率的なシステムに対するくさびのカタストロフ（*stochastic Cusp Catastrophe*，以下 *SCC* と呼ぶ．）を次のように定義している．

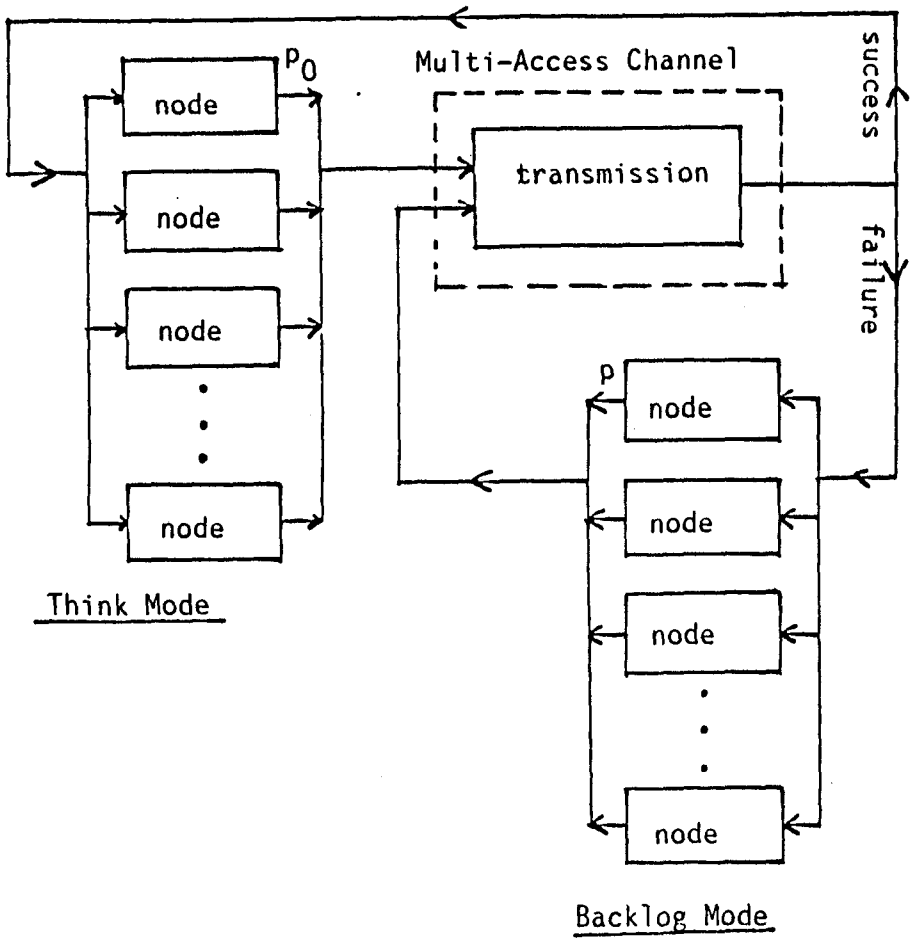


図 3.2: *slotted-ALOHA* 方式の解析モデル

定義５：ＳＣＣ

２つのコントロールパラメータをもつ確率的なシステムのポテンシャル関数を$V(x;c_1,c_2)$と表記する．$V(x;c_1,c_2)$を平衡状態の順序に注目して場合分けした時，$P(x;a,b)$と同様に５通りに場合分けされたならば，そのシステムは確率的くさびのカタストロフになる，という．

つまり，$V(x;c_1,c_2)$が$P(x;a,b)$と同じような形状変化を起こせば，確率的なシステムにおいて，$P(x;a,b)$と同じようなカタストロフが起こることが予想される，ということを意味している．

確率的なシステムの *Cusp* は$P'(x;a,b)=0$，$P''(x;a,b)$に対応する下記の２つの式を解いて得られる点(c_1,c_2)の集合として得ることができる..

$$V'(x;c_1,c_2) = 0 \tag{3.21}$$

$$V''(x;c_1,c_2) = 0 \tag{3.22}$$

(3.17)式より，(3.21)式，(3.22)式は次の式になる．

$$\mu(x;c_1,c_2) = 0 \tag{3.23}$$

$$\mu'(x;c_1,c_2) = 0 \tag{3.24}$$

slotted-ALOHA 方式のポテンシャル関数$V(x;p_0,p)$を図3.3に示す．この図より，図3.3aは$P(x;a,b)$の場合分けにおいて(2)に対応し，図3.3bは(5)および図3.3cは(4)に対応していることがわかる．また，(1),(3)に対する$V(x;p_0,p)$の形状もp，p_0の値を変えることに示すことができる．よって，*slotted-ALOHA* 方式が（確率的）くさびのカタストロフになることがわかる．

式(3.23)，式(3.24)より得られる *Cusp* を図3.4に示す．

確率的なシステムの性能と *Cusp* がどのように関連しているか，ということを明らかにするために，平均バックログと *Cusp* の関係を図3.5に示す．この図より，次のようなことがわかる．

- 点(p_0,p)が *Cusp* の内部を横切るように変化したとき，平均バックログは急激に変化する．
- 点(p_0,p)が *Cusp* の外部において変化するときは，平均バックログはゆるやかに変化する．

このように，パケットの到着率p_0が増加してきて，点(p_0,p)が *Cusp* を横切ったとき，平均バックログは急激に増加する．平均バックログの急激な増加は平均パケット

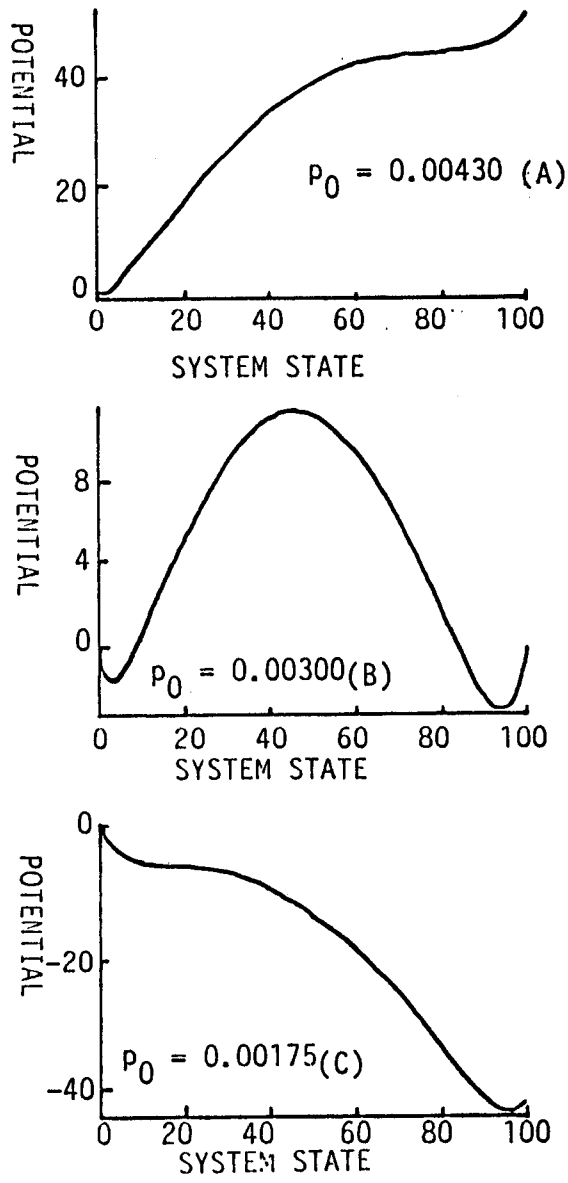


図 3.3: ポテンシャル関数の形状変化（*slotted-ALOHA*）

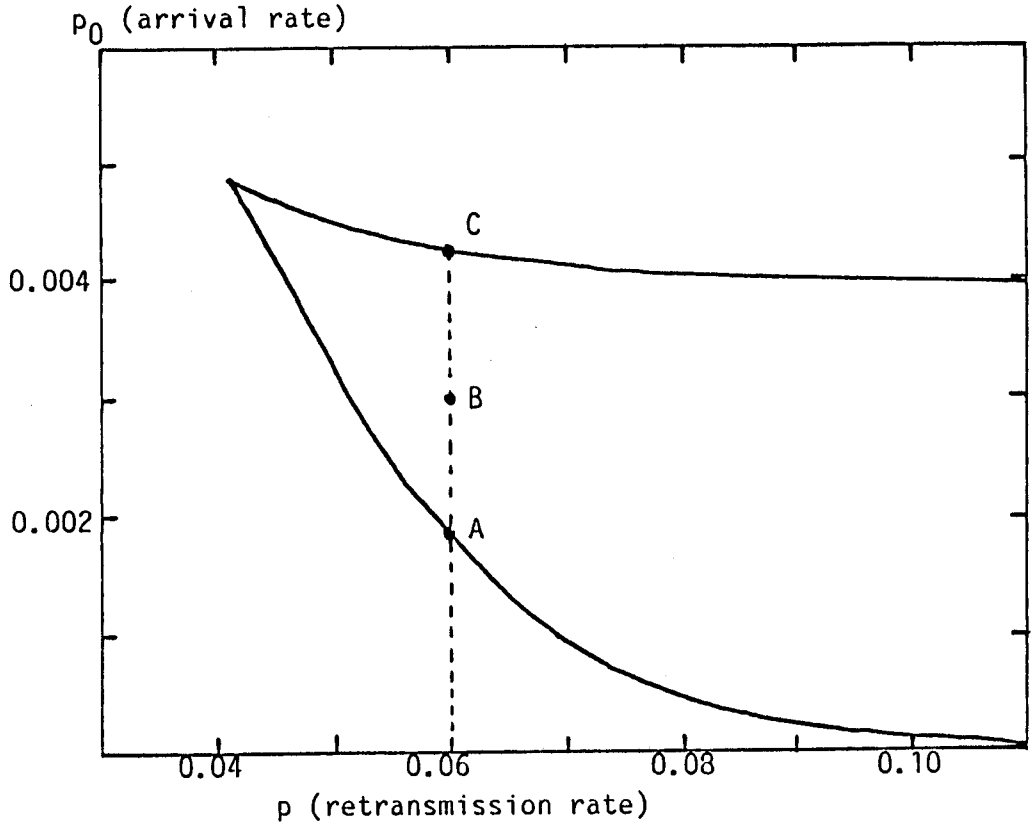


図 3.4: *Cusp (slotted-ALOHA)*

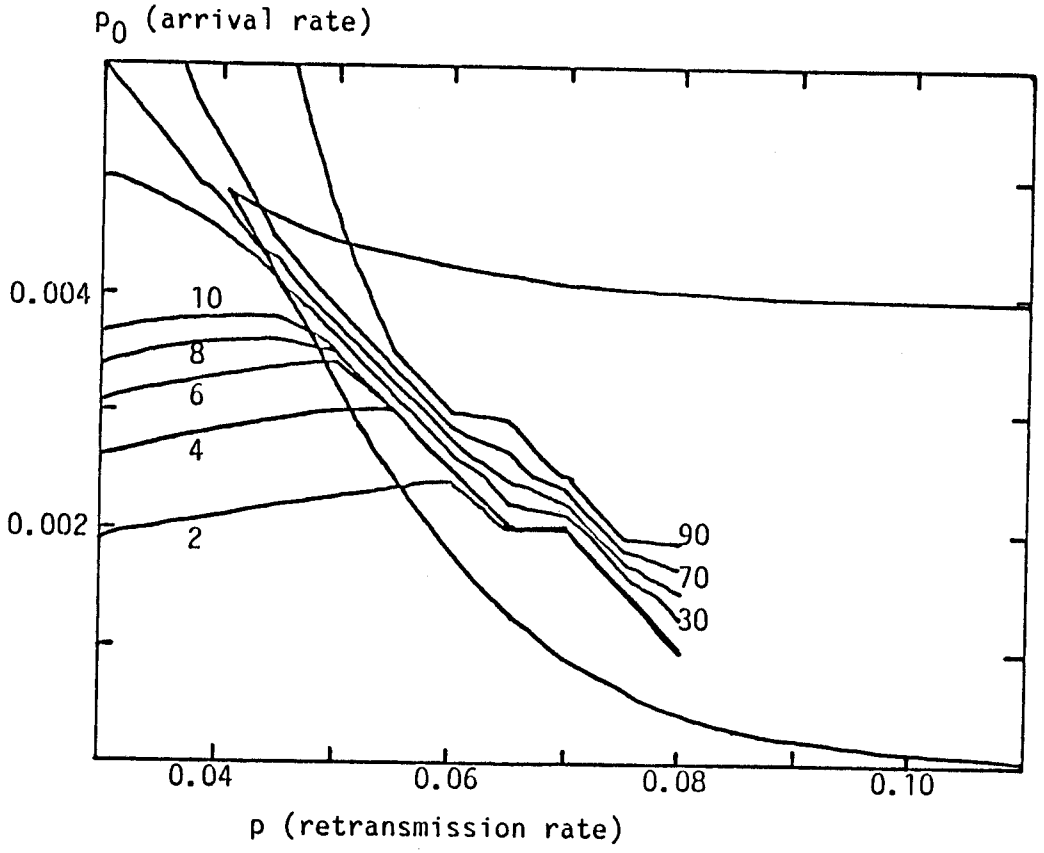


図 3.5: 平均バックログと *Cusp*

伝送遅延の急激な増加を意味する．よって，*Cusp* はシステムの性能が悪化する領域を示している，ということがわかる．

前章で述べたくさびのカタストロフにおいては，システムの状態は確率 1 でポテンシャル関数の極小値で安定する（静止している）ため，状態の急激な変化は *Cusp* の両側の曲線上（*Cusp* を完全に横切ったとき）で起こる．しかし平均バックログは，確率の重み和なので，コントロールパラメータが *Cusp* を完全に横切る前に急激な変化は起こる．

3.3 p-persistent CSMA/CD 方式の性能評価

3.3.1 p-persistent CSMA/CD 方式

CSMA/CD 方式は，主にバス型のローカル・エリア・ネットワークにおいて用いられるアクセス方式であり，パケットの送信を行う前に，既に送信中のパケットがチャネル上に存在するかどうかを調べ（*Carrier Sense*：送信中のパケットの有無を調べることをチャネルをセンスすると言う），また，チャネル上でパケットの衝突が起こったことを各ノードが検出できる（*Collision Detection*）といったような機構を備えた方式である．このように *CSMA/CD* 方式はチャネルのセンス，衝突の検出といった *ALOHA* 方式にはない機構を有しているので，*ALOHA* 方式よりも伝送効率の点で優れている．

p-persistent CSMA/CD 方式は *CSMA/CD* 方式の一つであり，次の手順に従ってパケットを送信する．

(1) 最初に，各ノードはパケットの送信要求を待っているモード（*Think* モード）にある．パケットの送信要求が発生したとき手続き(2)を行う．

(2) チャネルをセンスし，チャネルが空の状態（以下，*idle* と呼ぶ）の時，パケットの送信を行い，送信が成功した場合は(1)へ，送信が失敗した（チャネル上でパケット同士の衝突が起こった）場合は手続き(4)へ．チャネル上にパケットがある（以下，*busy* と呼ぶ）時，チャネルが *idle* になるまで待ち，*idle* になったら(3)の手続きへ．

(3)a 確率 p で送信を行い，送信が成功した場合は(1)へ，送信が失敗した場合は手続き(4)へ．

(3)b 確率$(1-p)$ で最大伝播遅延時間（最も離れたノード間で信号が伝播するのに

要する時間）だけ待って，再びチャネルをセンスし，*idle* ならば再び(3)の手続きを行い，*busy* ならば手続き(4)へ．

(4) ある時間間隔（バックオフ期間）だけ，パケットの送信を延期し，その時間が終了した時手続き(2)へ．

ここで，手続き(3)b，(4)を行ったノードは送信が成功するまで，*Backlog* モードにあると呼ぶ．

3.3.2 解析モデル

p-persistent CSMA/CD の解析を簡単にするために，以下のことを仮定する．

- 最大伝播遅延時間をスロットと呼び，各ノードはスロットに同期して，チャネルのセンス，パケットの送信を行う．
- パケット長は固定で，その送信時間は T スロットとする．
- ノード数は，有限数(M)で，各ノードは1パケット分のバッファをもっている．
- パケットを持っているノードはそれ以上のパケットは発生しない．
- 衝突を起こしたノードが送信を中止するまでの時間を γ スロットとする．
- 衝突発生時のバックオフ期間の確率分布を平均 $1/p$ の幾何分布とする．

この仮定に基ずいた *p-persistent CSMA/CD* の近似モデルを図 3.6 に示す．

- *Think* モードにあるノードは各スロット開始時に，確率 p_0 で新しいパケットを発生しチャネルをセンスする．チャネルが *idle* ならばパケットを送信し，*busy* ならば *Backlog* モードに入る．
- *Backlog* モードにあるノードは各スロット開始時に，確率 p でチャネルをセンスし，確率$(1-p)$ で，チャネルへのセンスを次のスロットまで待つ．チャネルをセンスした結果，チャネルが *idle* ならばパケットを再送信し，*busy* ならば *Backlog* モードに戻る．
- パケットの送信が失敗したノードは *Backlog* モードに入り，送信が成功したノードは *Think* モードに入る．

以下の説明において，確率 p_0 を到着率，確率 p を再送率と呼び，*Backlog* モードにあるノードの総数を *backlog* と呼ぶ．

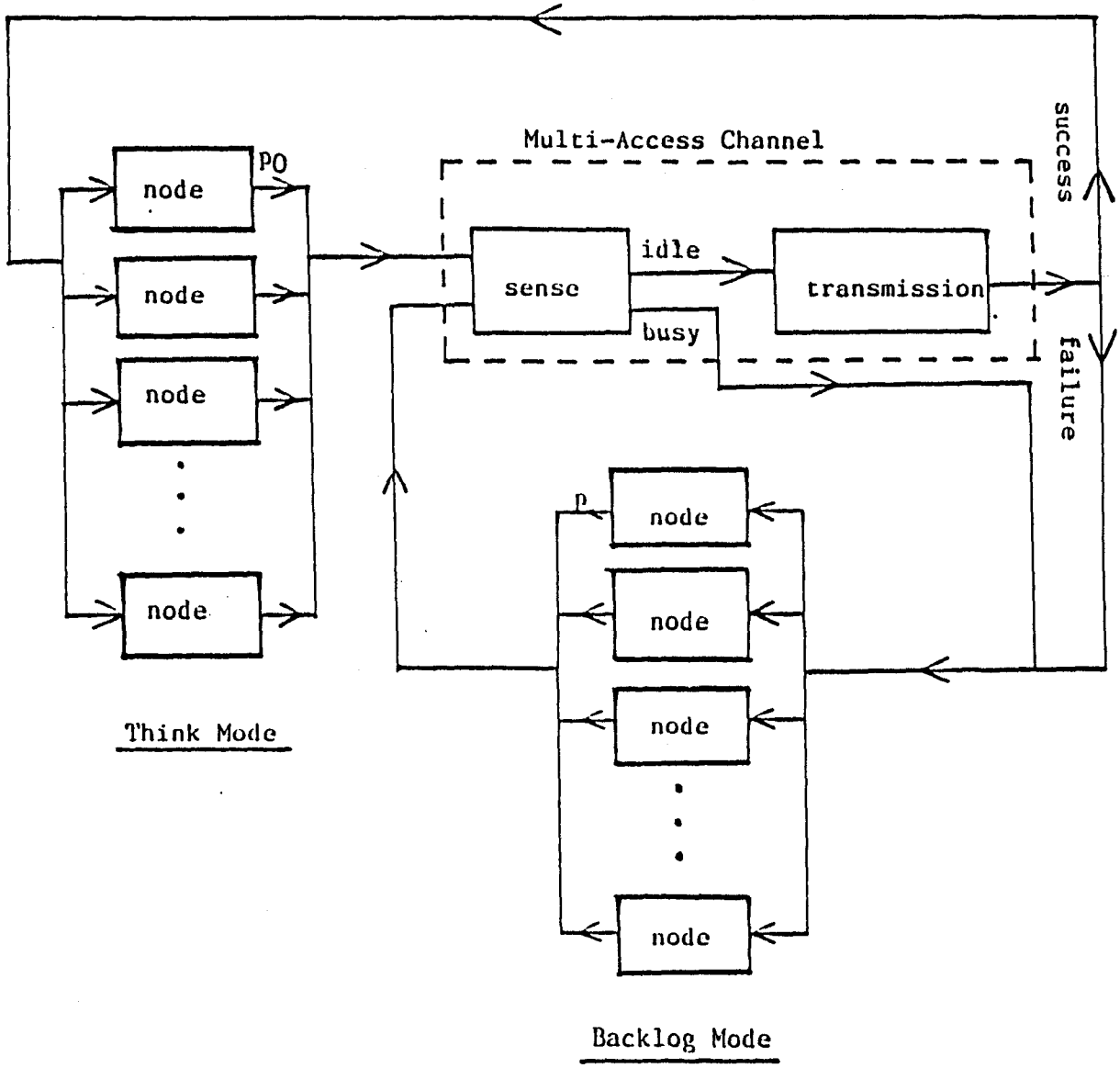


図 3.6: *p-persisten CSMA/CD* 方式の解析モデル

3.3.3 ポテンシャル関数

p-persistent CSMA/CD のシステム状態を，[9,10,11,12] と同じように，*backlog* とし，それを x で表わす．

まず，*p-persistent CSMA/CD* の定常状態における無限小平均を $\mu(x;p_0,p)$，無限小分散 $\sigma^2(x;p_0,p)$ を導出する[23]．

チャネルが *busy* から *idle* になった瞬間から次に *busy* から *idle* になるまでの期間を *cycle* と定義する．図 3.7 は *Backlog* モードのノードが送信したパケットが成功し，そのパケットの送信中に 4 つのノードがチャネルをセンスした状況を示す．

また，あるノードからのパケットの送信成功または送信失敗を他のすべてのノードが認識するために最大 1 スロットのずれがある．よってこのための時間 1 スロットを加算して解析を行う．

cycle 開始時の *backlog* を x とする．また仮定より，*cycle* 内のアイドル時間 I の分布はパラメータ P_{send} の幾何分布となる．

$$\begin{aligned} P_{send} &= Prob[\text{少なくとも 1 つのノードが送信する}] \\ &= 1-(1-p_0)^{(M-x)}(1-p)^x \end{aligned} \tag{3.25}$$

よって，アイドル時間の平均 $\bar{I}$，分散 σ_I^2 は次のようになる．

$$\bar{I} = 1/P_{send} \tag{3.26}$$

$$\sigma_I^2 = (1-P_{send})/P_{send}^2 \tag{3.27}$$

アイドル時間が終了するとつぎの(a)～(d)のうちいずれかの事象が生起する．(図 3.8 参照)

(a) *Think* モードからのパケットが成功する．
(b) *Backlog* モードからのパケットが成功する．
(c) *Backlog* モードからのパケット同志で衝突する．
(d) *Think* モードからのパケットが衝突に参加する．

(a),(b),(c),(d)のおこる確率を P_{ST}，P_{SB}，P_{CB} および P_{CT} とすると，それぞれは次のように求まる．

$$P_{ST} = (M-x)p_0(1-p_0)^{M-x-1}(1-p)^x/P_{send} \tag{3.28}$$

$$P_{SB} = xp(1-p)^{x-1}(1-p_0)^{M-x}/P_{send} \tag{3.29}$$

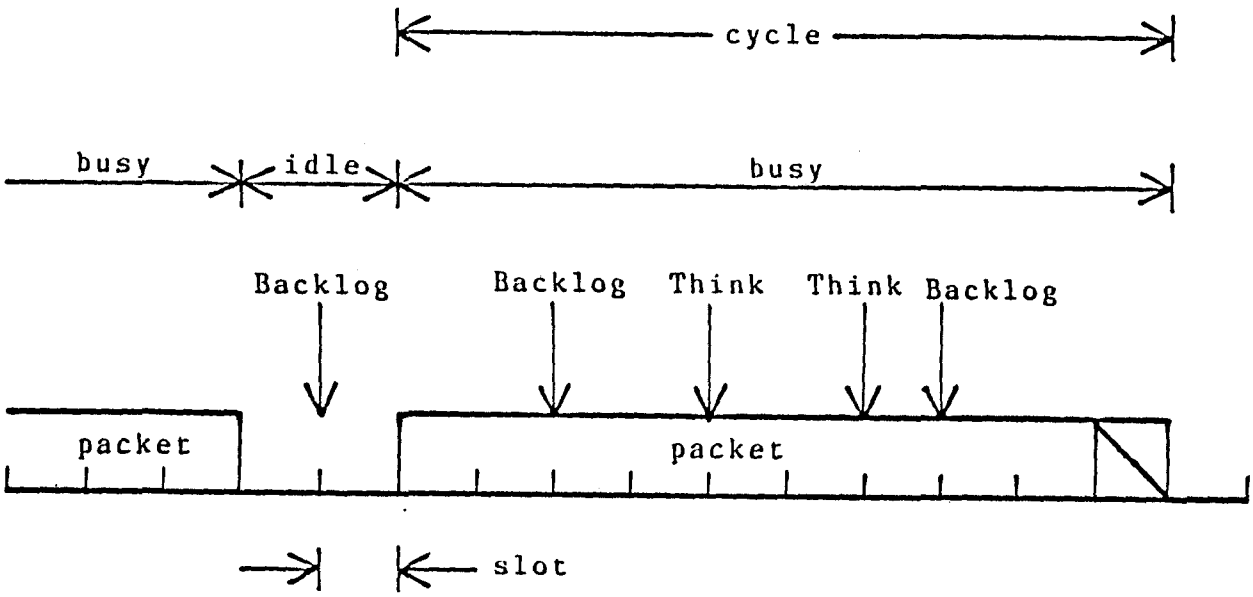
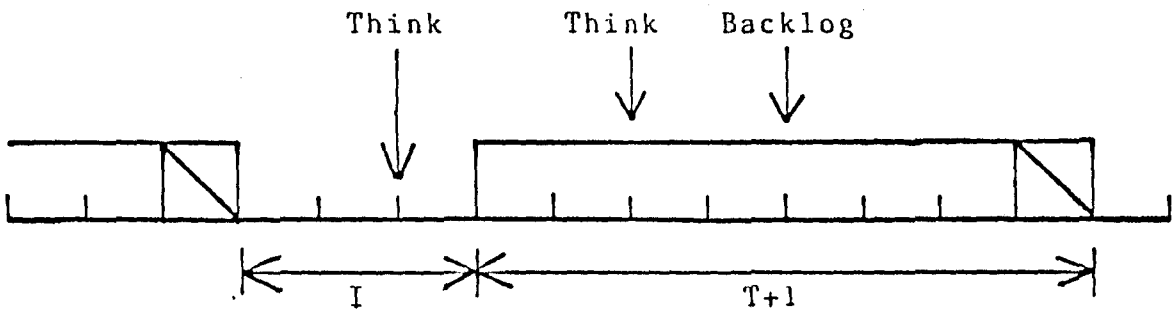
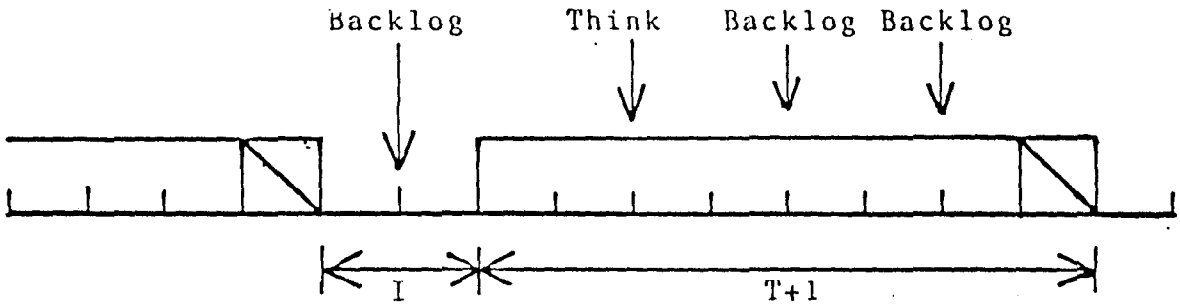

cycle
busy
idle
busy
Backlog
Backlog
Think
Think
Backlog
packet
packet
slot

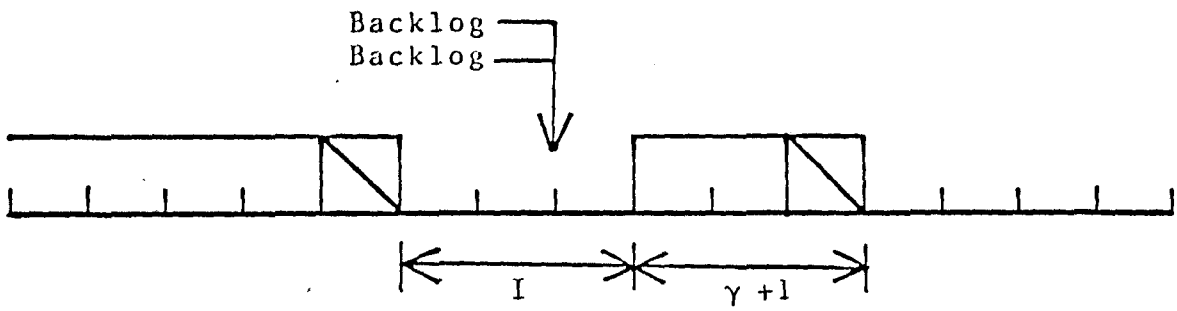
図 3.7: *cycle*



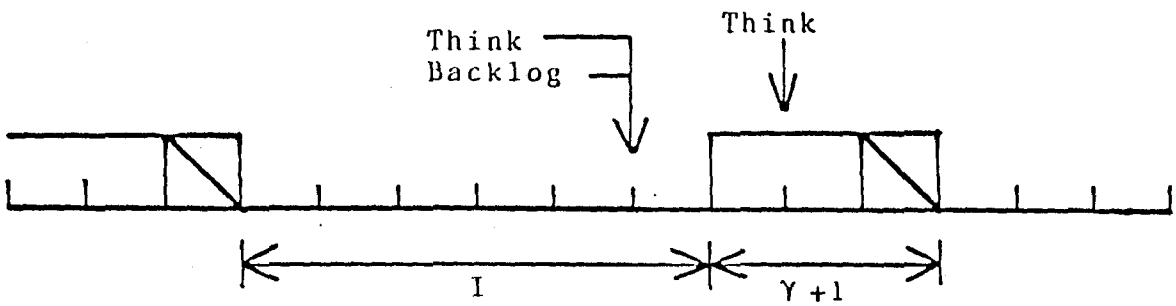
(a) success of packet in Think Mode



(b) success of packet in Backlog Mode



(c) collision without packet in Think Mode



(d) collision with packet in Think Mode

図 3.8: *cycle* 終了時に生起する事象

$$P_{CB} = (1-p_0)^{M-x}\{1-(1-p)^x - xp(1-p)^{x-1}\}/P_{send} \quad (3.30)$$

$$P_{CT} = 1-(P_{ST}+P_{SB}+P_{CB}) \quad (3.31)$$

また，送信が成功，失敗した時の平均 *cycle* 長 C_S, C_F は，パケット長 T および衝突を起こしたノードが送信を中止するのに要する時間 γ を用いて，次のように求まる．

$$C_S = \bar{I}+T+1 \quad (3.32)$$

$$C_F = \bar{I}+\gamma+1 \quad (3.33)$$

よって平均 *cycle* 長 $\bar{C}$ は

$$\bar{C} = (P_{ST}+P_{SB})C_S + (P_{CB}+P_{CT})C_F \quad (3.34)$$

いま，*backlog* が減少する過程（待ち行列システムにおいて退去過程に対応する）に注目する．単位時間におけるパケットの送信成功率 $S(packets/slot)$ は

$$S = (P_{ST}+P_{SB})/\bar{C} \quad (3.35)$$

とくに，*Backlog* モードからのパケット成功率 S_B は

$$S_B = P_{SB}/\bar{C} \quad (3.36)$$

よって退去間隔（*backlog* が 1 だけ減少するのに要するスロット数）M_{dB} は

$$M_{dB} = 1/S_B = \bar{C}/P_{SB} \quad (3.37)$$

また，パケットの送信成功間隔の分散 σ_d^2 は

$$\sigma_d^2 = \{P_S^2\sigma_S^2 + P_S(1-P_S)\sigma_F^2 + (1-P_S)C_F^2\}/P_S^2 \quad (3.38)$$

ただし，$P_S = P_{ST}+P_{SB}$, σ_S^2 および σ_F^2 はそれぞれ送信が成功，失敗した時の *cycle* 長の分散であり，$\sigma_S^2 = \sigma_F^2 = \sigma_I^2$ となる．とくに *Backlog* モードからのパケットの送信成功間隔の分散 σ_{dB}^2 は，送信成功したパケットが *Backlog* モードのノードから送信されたものである確率 $P_B = P_{SB}/P_S$ を用いて次のようになる．

$$\sigma_{dB}^2 = \sigma_d^2/P_B + (1-P_B)/(SP_B^2) \quad (3.39)$$

次に *backlog* が増加する過程（待ち行列システムにおいて到着過程に対応する）に注目する．まずパケットが送信成功の場合，$(T+1)$ スロットの間に *Think* モードの

ノードからアクセスされた回数分だけ *backlog* は増加する．また衝突時には衝突を起こした *Think* モードのノードに加えて$(\gamma+1)$スロットの間にアクセスした *Think* モードのノードが新たに *Backlog* モードに入る．これらのことを考慮して，前に述べた(a)～ (d)の場合の *backlog* の増加量 A_{ST}, A_{SB}, A_{CB} および A_{CT} はそれぞれ次のようになる．

$$A_{ST} = (M-x-1)p_0(T+1) \tag{3.40}$$
$$A_{SB} = (M-x)p_0(T+1) \tag{3.41}$$
$$A_{CB} = (M-x)p_0(\gamma+1) \tag{3.42}$$
$$A_{CT} = 1+(M-x-1)p_0(\gamma+1) \tag{3.43}$$

よって *backlog* の平均増加率$\bar{A}$ は

$$A = (P_{ST}A_{ST}+P_{SB}A_{SB}+P_{CB}A_{CB}+P_{CT}A_{CT})/\bar{C} \tag{3.44}$$

この増加率を到着率と考え，到着過程をポアソン分布とみなした場合，到着間隔の平均 M_a, 分散σ_a^2 は

$$M_a = 1/\bar{A} \tag{3.45}$$
$$\sigma_a^2 = 1/\bar{A}^2 \tag{3.46}$$

いま求めた M_a, M_{dB}, σ_a^2 およびσ_{dB}^2 を用いて無限小平均$\mu(x;p_0,p)$ ，無限小分散 $\sigma^2(x;p_0,p)$ は次のようになる．

$$\mu(x;p_0,p) = 1/M_a - 1/M_{dB} \tag{3.47}$$
$$\sigma^2(x;p_0,p) = 1/M_a + \sigma_{dB}^2/M_{dB}^3 \tag{3.48}$$

いま導出した無限小平均，無限小分散を用いて，(3.17)式に従って，*p-persistent CSMA/CD* 方式のポテンシャル関数を *slotted-ALOHA* 方式と同様に次のように得ることができる．

$$V(x;p_0,p) = -\int_0^x \frac{2\mu(x;p_0,p)}{\sigma^2(x;p_0,p)}dx \tag{3.49}$$

図 3.9 に *p-persistent CSMA/CD* 方式のポテンシャル関数の典型的な形状を示す$(p=0.08$ および $p_0=0.00012)$．尚，以下で示す数値例は，特別に言及しない限り $M=100$, $T=100$, $\gamma=2$ の場合である．図 3.9 の点A，点B および点C において，ポテンシャル関数の定義より，ドリフト関数の値は 0 である．よって，ドリフト関数の定義より，これらの点での無限小平均は 0 である．無限小平均が 0 ということは，

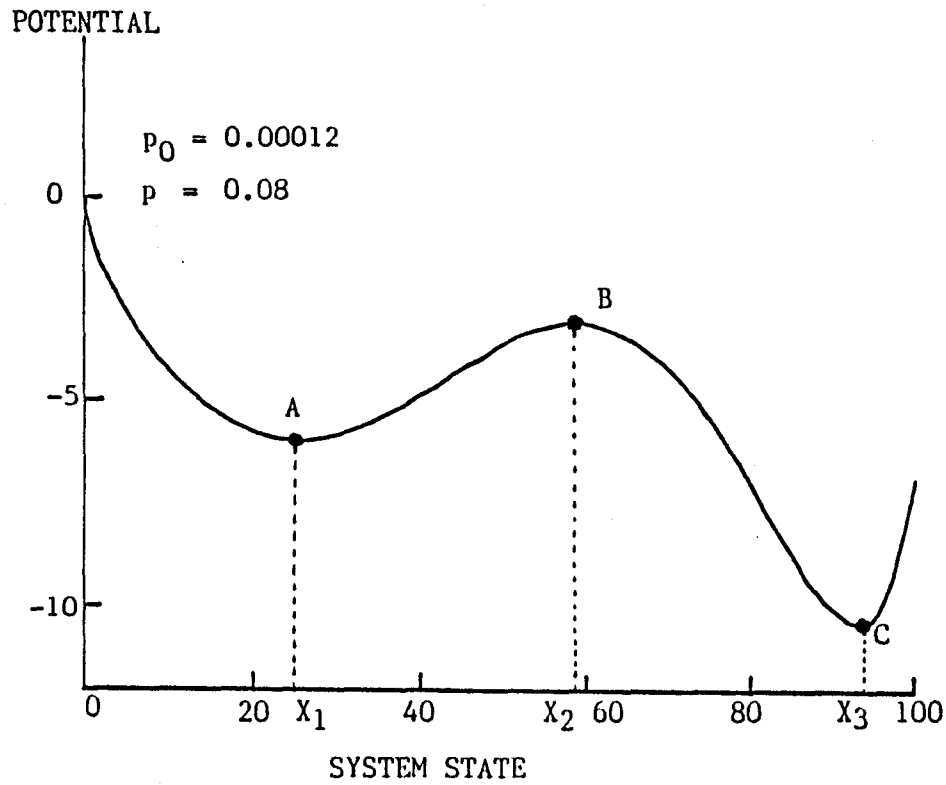


図 3.9: ポテンシャル関数（*p-persistent CSMA/CD*）

その状態において，システムに対して入って来るパケットの数と出ていくパケットの数が等しいことを意味する．よって，状態 x_1，x_2 および x_3 は平衡状態になっている．また，ドリフト関数の定義より，状態 x_1 の負側の状態では，システムに入って来るパケット数は出ていくパケットの数より大きく，逆に正側の状態では，出ていくパケットの数が大きい．このようにシステムは状態 x_1 で安定しようと動作する．よって，状態 x_1 は安定平衡状態である．同様に，状態 x_3 も安定平衡状態になる．逆に，状態 x_2 は不安定平衡状態である．

3.3.4 Cusp

パケットの再送率 p の値を固定して，パケットの到着率 p_0 の値が変化したときのポテンシャル関数 $V(x;p_0,p)$ を図 3.10 に示す．

図 3.10a，図 3.10b，図 3.10c，図 3.10d および図 3.10e は，平衡状態の順序がそれぞれ，(安定)，(安定→退化)，(安定→不安定→安定)，(退化→安定) および (安定) の例であり，$P(x;a,b)$ の形状の場合分けにおける(1)，(2)，(5)，(4) および(1)の場合の形状に対応する．また，$P(x;a,b)$ の形状の(3)の場合分けに対応する $V(x;p_0,p)$ の形状は(p_0,p) の値を変えることによって示すことができる．よって，*p-persistent CSMA/CD* 方式はコントロールパラメータとして到着率 p_0，再送率 p を持つくさびのカタストロフになる，ということがわかる．

式(3.23)，式(3.24)より決定される *Cusp* を図 3.11 に示す．図 3.10，図 3.11における$(p_0,p)=(0.00014,0.075)$，$(0.0001369,0.075)$，$(0.000128,0.075)$，$(0.000121,0.075)$ および$(0.00011,0.075)$ は，それぞれ *Cusp* の上側外部の点（A点），上側の曲線上の点（B点），*Cusp* の内部の点（C点），*Cusp* の下側の曲線上の点（D点）および下側外部の点（E点）である．このように *p-persitent CSMA/CD* のポテンシャル関数の形状は，$P(x;a,b)$ と同様に，p_0 が増加して，*Cusp* を横断するときに，単安定形状から双安定形状を経由して，単安定形状に移動している．よって，システムは *Cusp* の内側で双安定モード，*Cusp* の外側で単安定モードになっている．

拡散近似[24,25]を用いて，定常状態密度関数は次のように求まる．

$$f(x;p_0,p)=Cexp[-V(x;p_0,p)-ln[\sigma^2(x;p_0,p)]] \tag{3.50}$$

この式で C は *normalizing constant* である．図 3.12 に(p_0,p)の値がA点，C点およびE点のときの $f(x;p_0,p)$ を示す．図 3.12において，到着率が小さい(0.000121)とき，システムは *backlog* の小さい状態にある確率が大きいことがわかる．ところが，到着

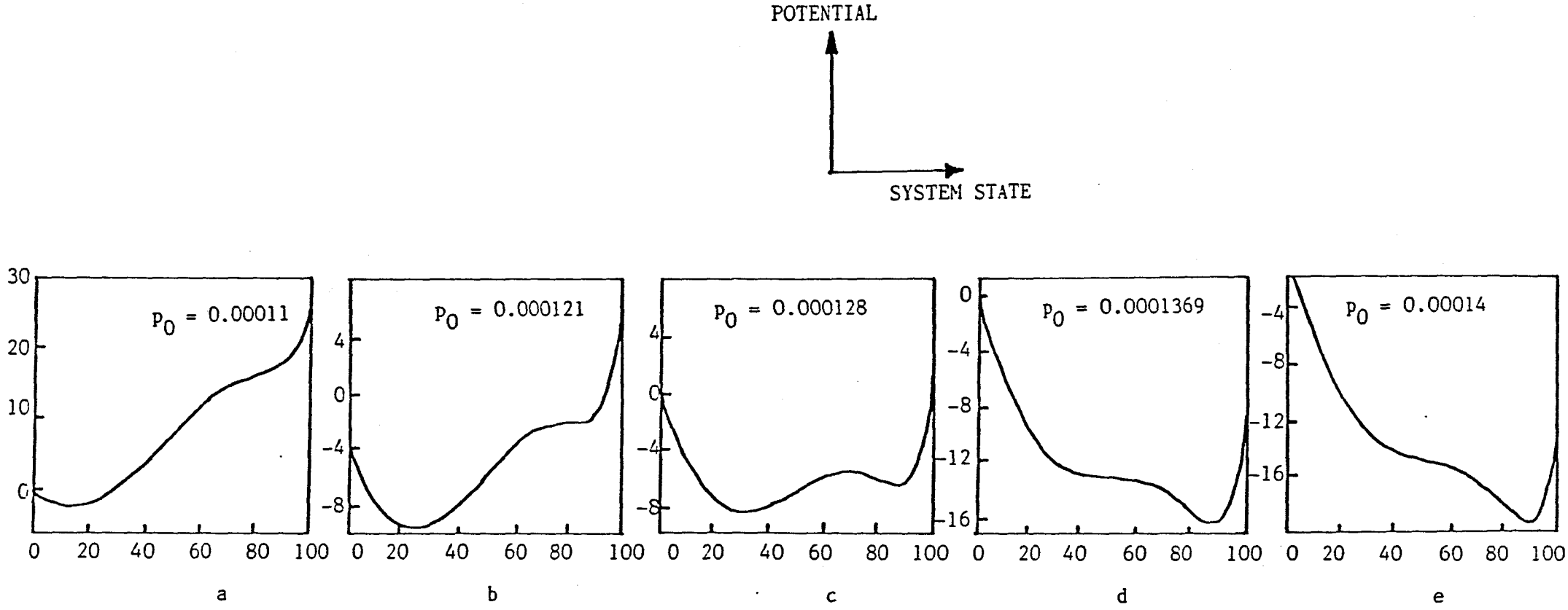


図 3.10: ポテンシャル関数の形状変化（*p-persistent CSMA/CD*）

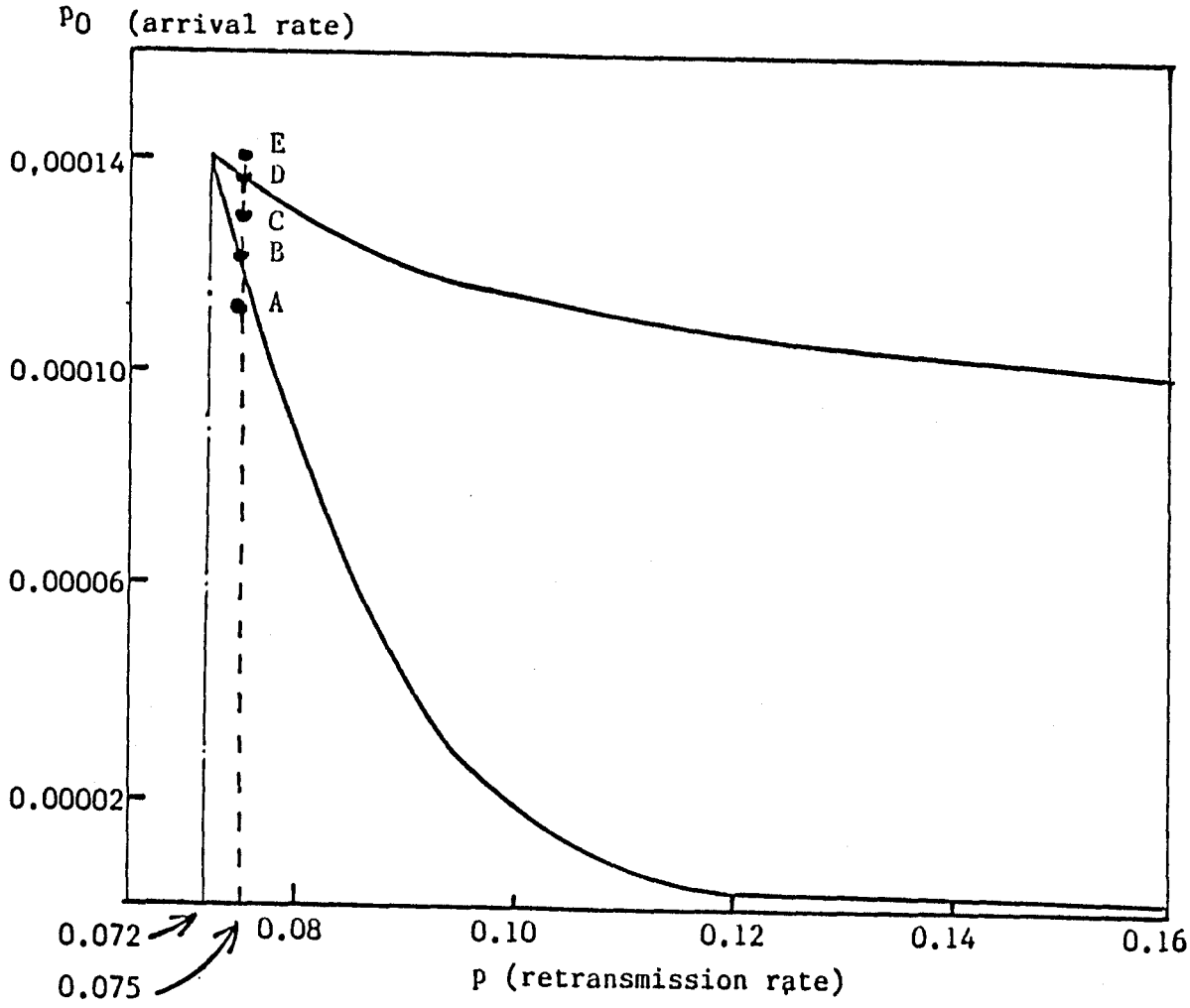
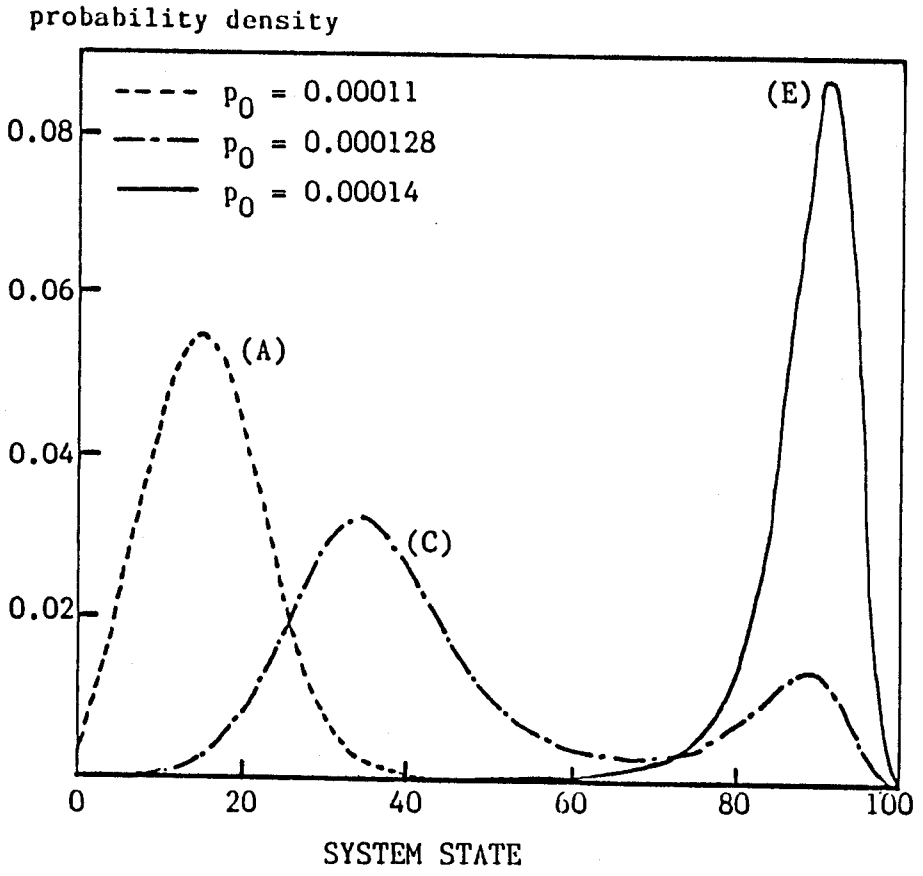


図 3.11: *Cusp (p-persistent CSMA/CD)*



(A) lower side of the cusp(outside the cusp)
(C) inside the cusp
(E) upper side of the cusp(outside the cusp)

図 3.12: 定常状態密度関数（ *p-persistent CSMA/CD* ）

率が大きくなってくると，*backlog* の小さい状態の確率は減少し，*backlog* の大きい状態の確率が増加し，さらに到着率が大きくなってくると，システムは *backlog* の大きい状態にある確率が大きくなることがわかる．また，ポテンシャル関数の極小値と定常状態密度関数の極大値を対応させ，ポテンシャル関数の極大値と定常状態密度関数の極小値を対応させればわかるように，ポテンシャル関数の形状変化と定常状態密度関数の形状変化は *Cusp* に対して同じになることがわかる．

3.3.5 平均バックログ

与えられた到着率，再送率に対して平均バックログ$\bar{x}$は先に求めた$f(x;p_0,p)$を用いて次の式より求まる．

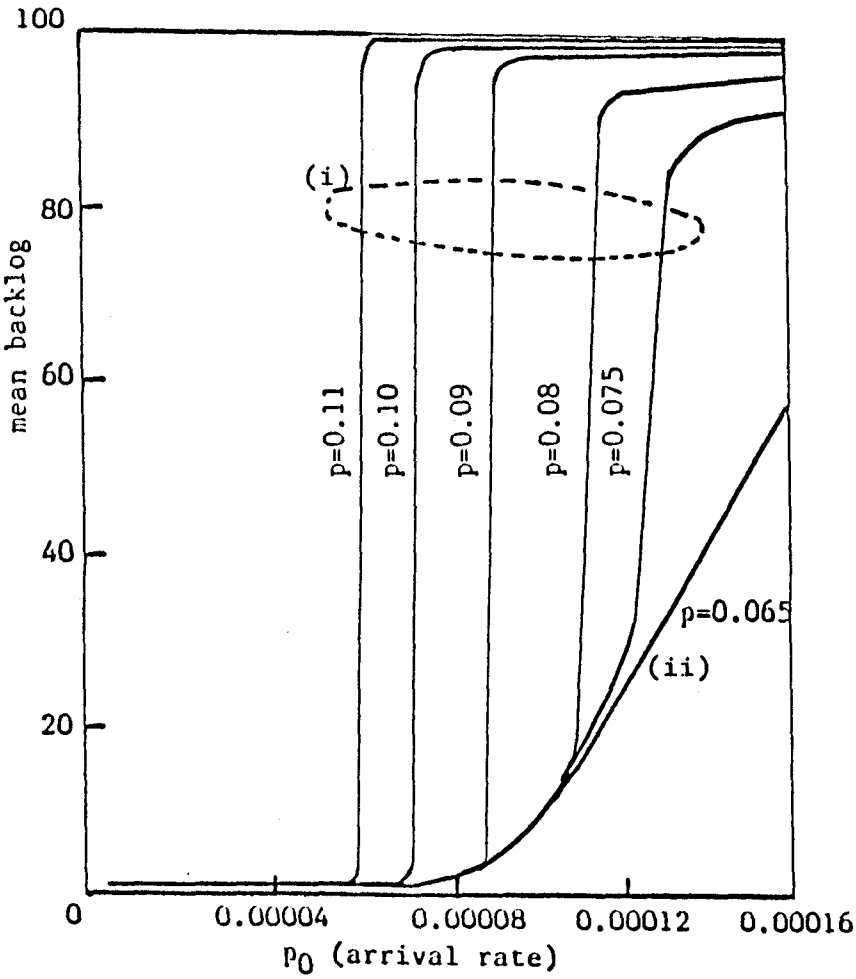
$$\bar{x} = \int_0^M x f(x;p_0,p)dx \tag{3.51}$$

再送率pを固定して，パケットの到着率p_0が徐々に大きくなった時（0.00001から0.00016）の平均バックログの変化を図3.13に示す．

図3.11の(p_0,p)平面において，再送率pが小さい時(0.072以下)には到着率がどのような値になろうと，点(p_0,p)は *cusp* の外側の点である．よって，再送率が0.072以下の時，到着率が0.00001から0.00016に徐々に増加していく時，システムは単安定モードのままであり，到着率に対して平均バックログもゆるやかに変化している．(図3.13において，$p=0.065$の曲線）ところが，再送率pが大きい時(0.072以上)には，到着率が増大するにつれて，図3.11の(p_0,p)平面において，点(p_0,p)は *Cusp* の下側から内側を経由して上側にでる．この時，システムのモードは単安定モード→双安定モード→単安定モードと移動し，平均バックログは急激に変化する．(例えば，図3.13において，$p=0.11$の曲線）つまり，*Cusp* の上側と下側の点(p_0,p)に対してシステムの安定モードは単安定モードであるが，*Cusp* の下側の点(p_0,p)に対しては平均バックログの小さい単安定モード（非飽和単安定モード），*Cusp* の上側の点(p_0,p)に対して平均バックログの大きい単安定（モード）になっている．今述べたように，決定性のシステムと同様に確率的なシステムにおいても，*Cusp* を横断するようにコントロールパラメータが変化した時カタストロフが起こる．

3.3.6 スループット，平均パケット伝送遅延

まず，スループットを求める．チャネルが *busy*（送信成功，送信失敗の両方を考慮）である時間に対して，送信が成功したパケットによってチャネルが占有された時間



(i) trajectory crosses over the cusp

(ii)trajectory doesn't cross over the cusp

図 3.13: 平均バックログ (*p-persistent CSMA/CD*)

の割合をスループットとする．式(3.35)よりスループット$\bar{S}$は次のようになる．

$$\bar{S} = \frac{\int_0^M f(x;p_0,p)(P_{ST}+P_{SB})Tdx}{\int_0^M f(x;p_0,p)\bar{C}dx} \tag{3.52}$$

次に，平均パケット伝送遅延（パケット伝送時間Tで正規化したもの）を求める．1 *cycle* あたりのチャネルノード数Nを，チャネルの *idle* 時に *Backlog* モードにあるノード数と，チャネルの *busy* 時にアクセスしたノード数(*Backlog* モード，*Think* モードも含めて）との和と定義する．Nの平均値$\bar{N}$は次のようになる．

$$\bar{N} = \int_0^M f(x;p_0,p)(x+\bar{A}\bar{C})dx \tag{3.53}$$

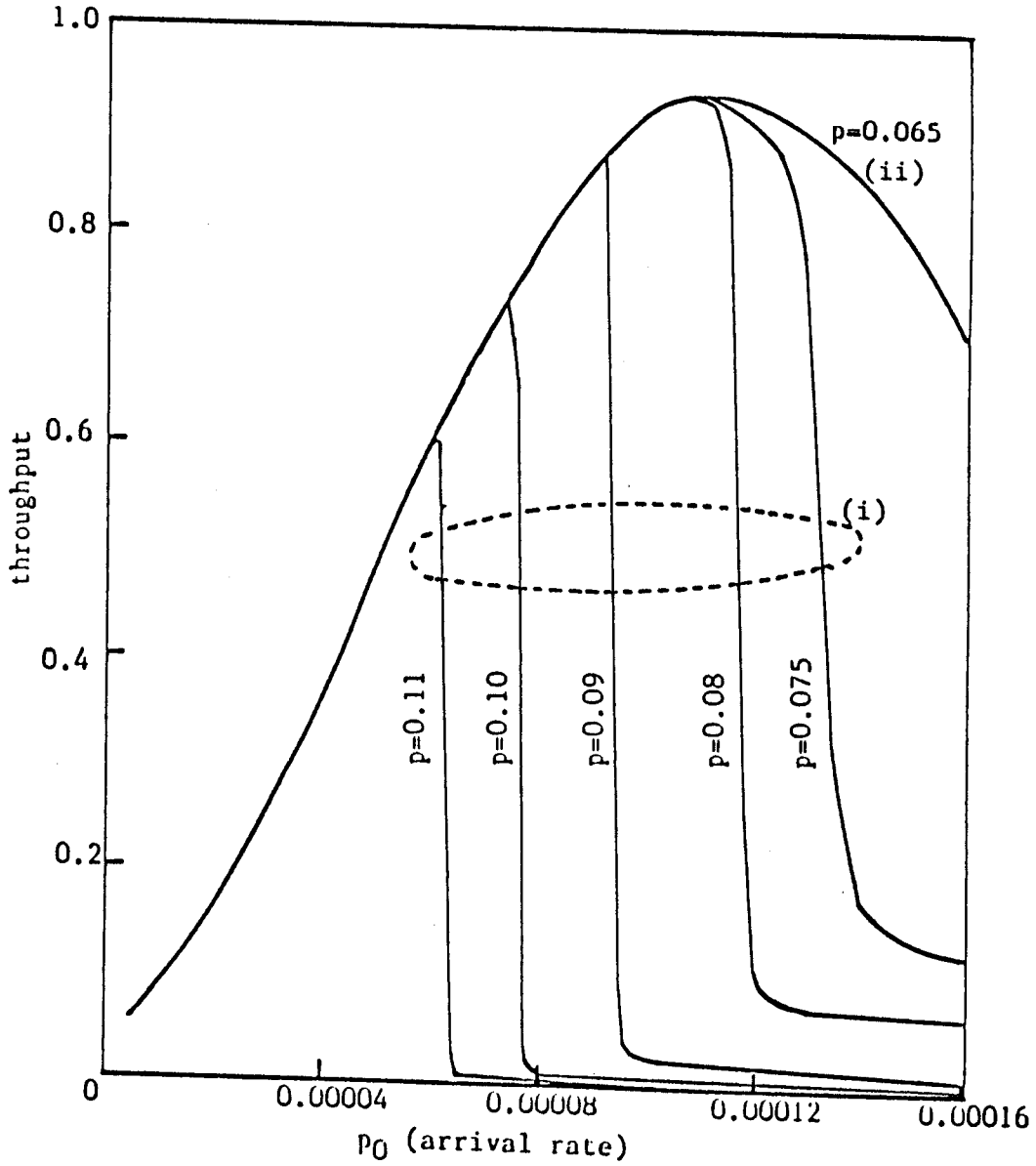
式(3.52)，式(3.53)によって得られる$\bar{S}$，$\bar{N}$を用いるとパケット伝送時間Tで正規化された平均パケット伝送遅延$\bar{D}$はリトルの公式[26]より次のようになる．

$$\bar{D} = \bar{N}/\bar{S} \tag{3.54}$$

(p_0,p)の値の変化に対するのスループット，平均パケット伝送遅延をそれぞれ図3.14，図3.15に示す．(p_0,p)の値が *Cusp* の内部に入るように変化する場合と，内部に入らないような変化をする場合の2つの場合におけるスループット，平均パケット伝送遅延の変化を考える．いま，再送率$p = 0.065$の場合，到着率の変化に対してスループット，平均パケット伝送遅延ともにゆるやかに変化している．ところが，例えば$p = 0.10$の場合，(p_0,p)の値が *Cusp* の内部を通過し，スループットは急激に小さくなり，平均パケット伝送遅延は急激に大きくなるといった急激な変化が起こっている．つまり，コントロールパラメータが *Cusp* の内部を通過する場合としない場合を比較した場合，平均バックログの変化と同じように，スループット，平均パケット伝送遅延の変化も，著しく異なっていることがわかる．

3.3.7 再送率による制御

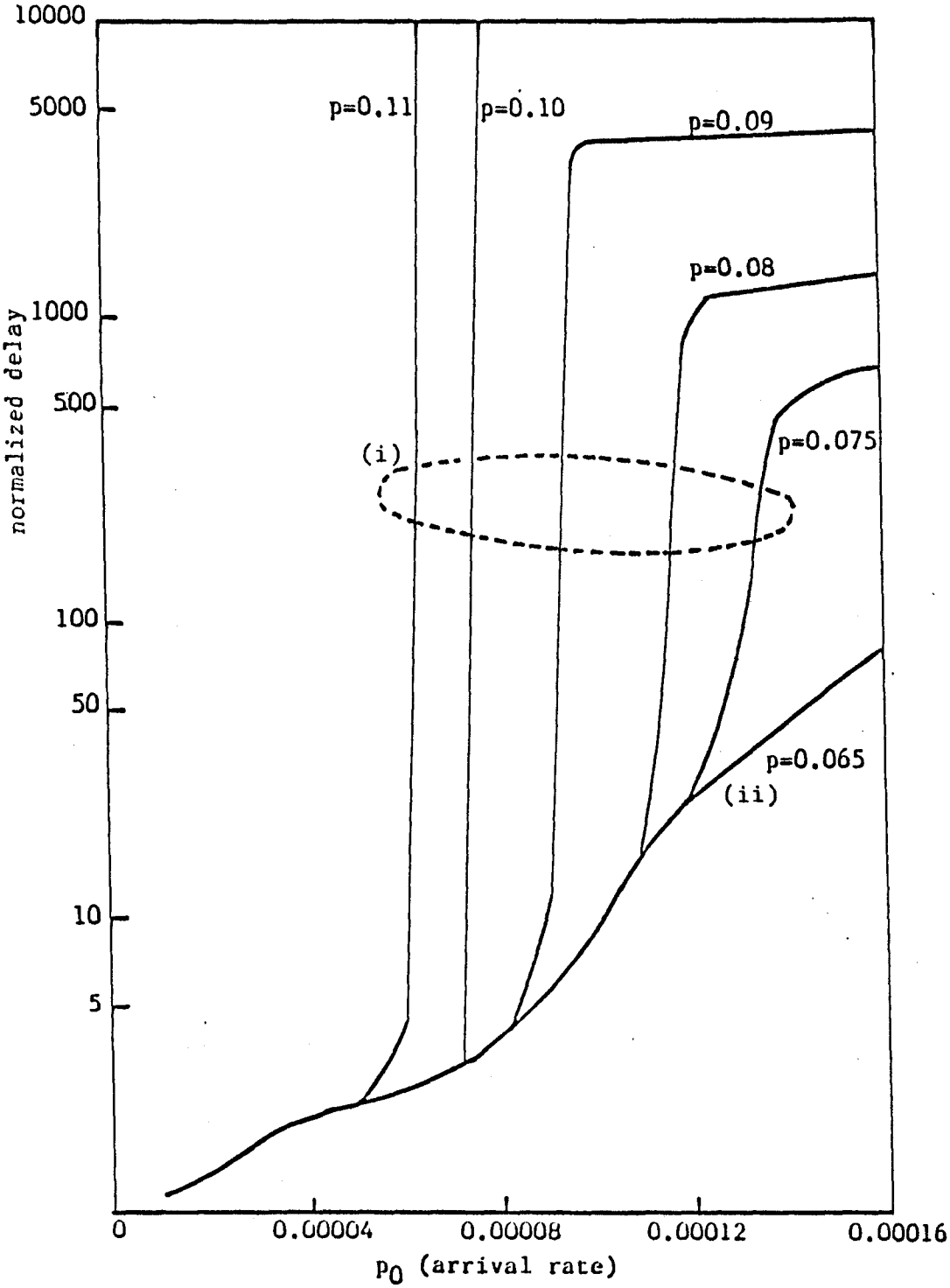
p-persistent CSMA/CD などいわゆるランダム・アクセス方式においては，システムの効率を高く維持するためには，トラヒックが少ない時には再送率を大きくし，トラヒックが多い時には再送率を小さくする必要がある[4]．ところが，ある時点の到着率に対して最適な再送率を決定することは困難であった．ここでは，*p-persistent CSMA/CD* において，ある時点の到着率に対して最適な再送率を決定する一方法として *Cusp* を用いる方法を述べる．いままで述べてきたようにシステムのコントロールパラメータが *Cusp* の内部を通過した場合，システムの効率は急激に悪化する．言い替えれば，コントロールパラメータの変化によって，システムが双安定モードを経



(i) trajectory crosses over the cusp

(ii)trajectory does't cross over the cusp

図 3.14: スループット (*p-persistent CSMA/CD*)



(i) trajectory crosses over the cusp

(ii)trajectory does't cross over the cusp

図 3.15: 平均パケット伝送遅延（ *p-persistent CSMA/CD* ）

験したとき，システムの効率は急激に悪化する．よって，到着率が増大した場合に，コントロールパラメータがが *Cusp* の内部に入らないように *Cusp* の下側の曲線に沿って，再送率をコントロールすればシステムの効率の急激な悪化を回避することができると考えられる．再送率に対して，このような制御を行った場合の平均バックログの変化を図 3.16 に示す．この図では，図 3.17 に示すように，到着率の増加に伴ってなんら制御を行わない（つまり，再送率を固定する）場合（$w1$）と，到着率の増加に伴って *Cusp* の内部に入らないように再送率を調整した場合（$w2$）の平均バックログを示している．図 3.16 からこのような制御を行えばシステムは双安定モードを経験しないので，平均バックログの急激な悪化をある程度回避できることがわかる．

3.3.8 Cuspの変化

これまでは，ノード数 M，パケット長 T を固定して議論を行なってきたが，ここでは，M，Tの値によって，*Cusp* がどのように変化するか，ということについて議論する．

まず，チャネルの負荷 G を次のように定義する．

$$G = MTp_0 \tag{3.55}$$

T とγ をそれぞれ 100，2 に固定して，M の値が変化したとき（$M = 200$，100，50）の *Cusp* の変化を(p, G) 平面に（図 3.18）に示す．M が増加することはパケット同士の衝突の確率が増加することを意味する．いったん衝突が起こったときに，再度衝突する確率を減らし，システム性能をゆるやかに変化させるためには，再送率は小さくなければならない．よって，M の増加に対して，*Cusp* は再送率 p 軸の負の方向に移動する．

次に，M とγ をそれぞれ 100，2 に固定して，T の値が変化したとき（$T = 100$，50，25）の *Cusp* の変化を(p, G) 平面に（図 3.19）に示す．G を固定したとき，T が減少するにつれて，G の定義より，p_0 が大きくなる．p_0 が大きくなることは，パケット同士の衝突の確率が増加することを意味する．よって，*Cusp* の M による変化と同様な理由で，T の減少に対して，*Cusp* は再送率 p 軸の負の方向に移動する．

M，T はシステム設計時に決定されるパラメータであり，動的に変化させることはできないが，図 3.18，図 3.19 を，システム設計時に，あらかじめ用意することにより，予想される到着率に対して，システム性能の急激な劣化を防ぐための M，T に対する情報を得ることができる．

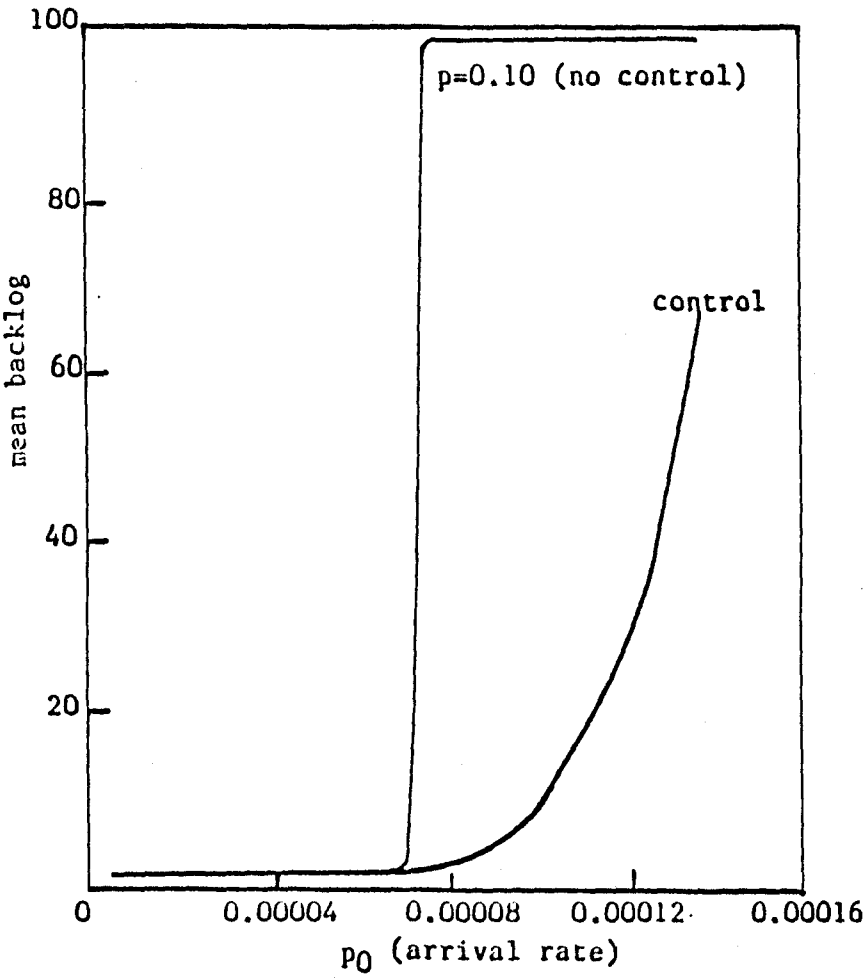


図 3.16: 再送率を制御した場合の平均バックログ

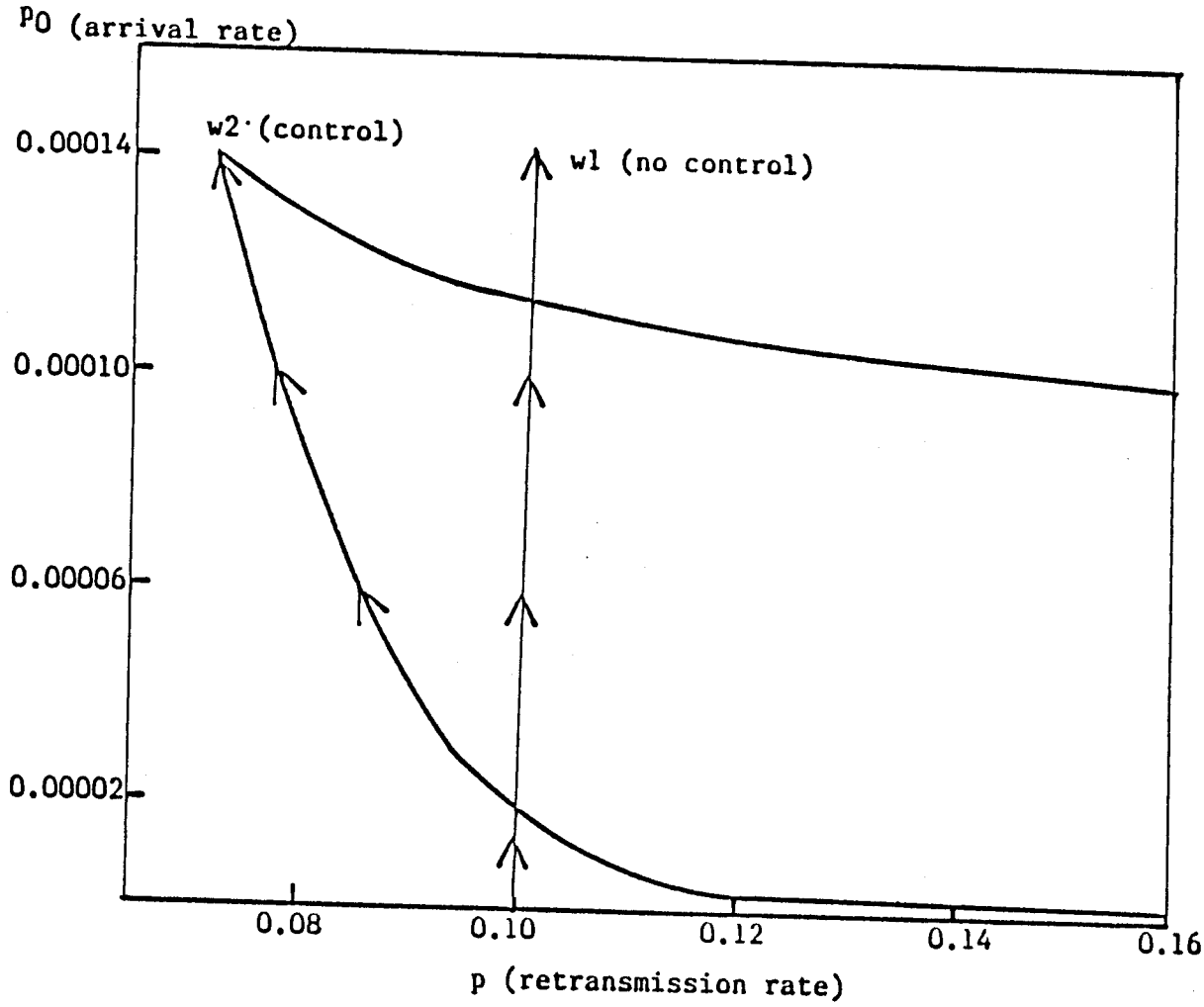


図 3.17: コントロールパラメータの軌跡

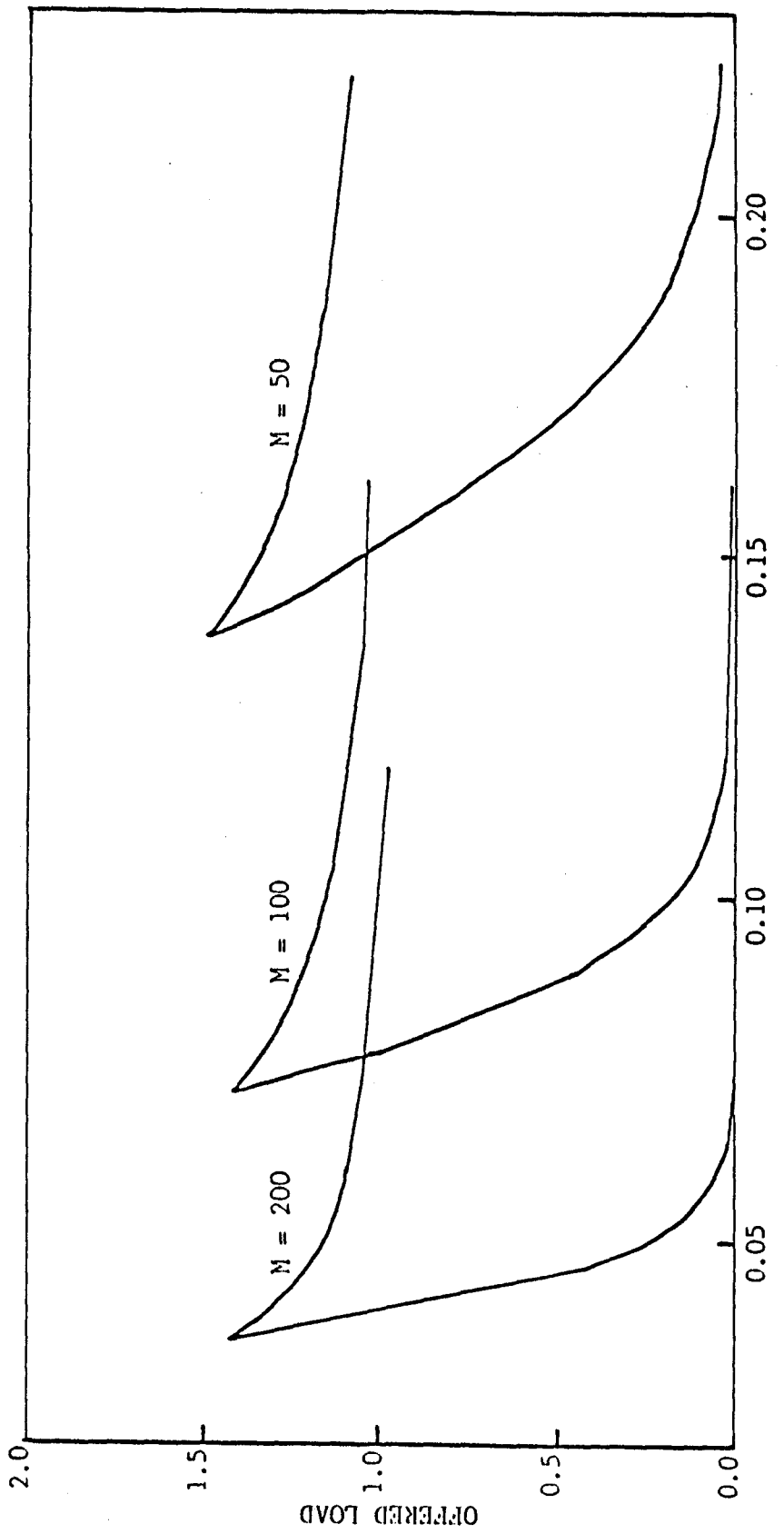


図 3.18: M が変わったときの $Cusp$ の変化

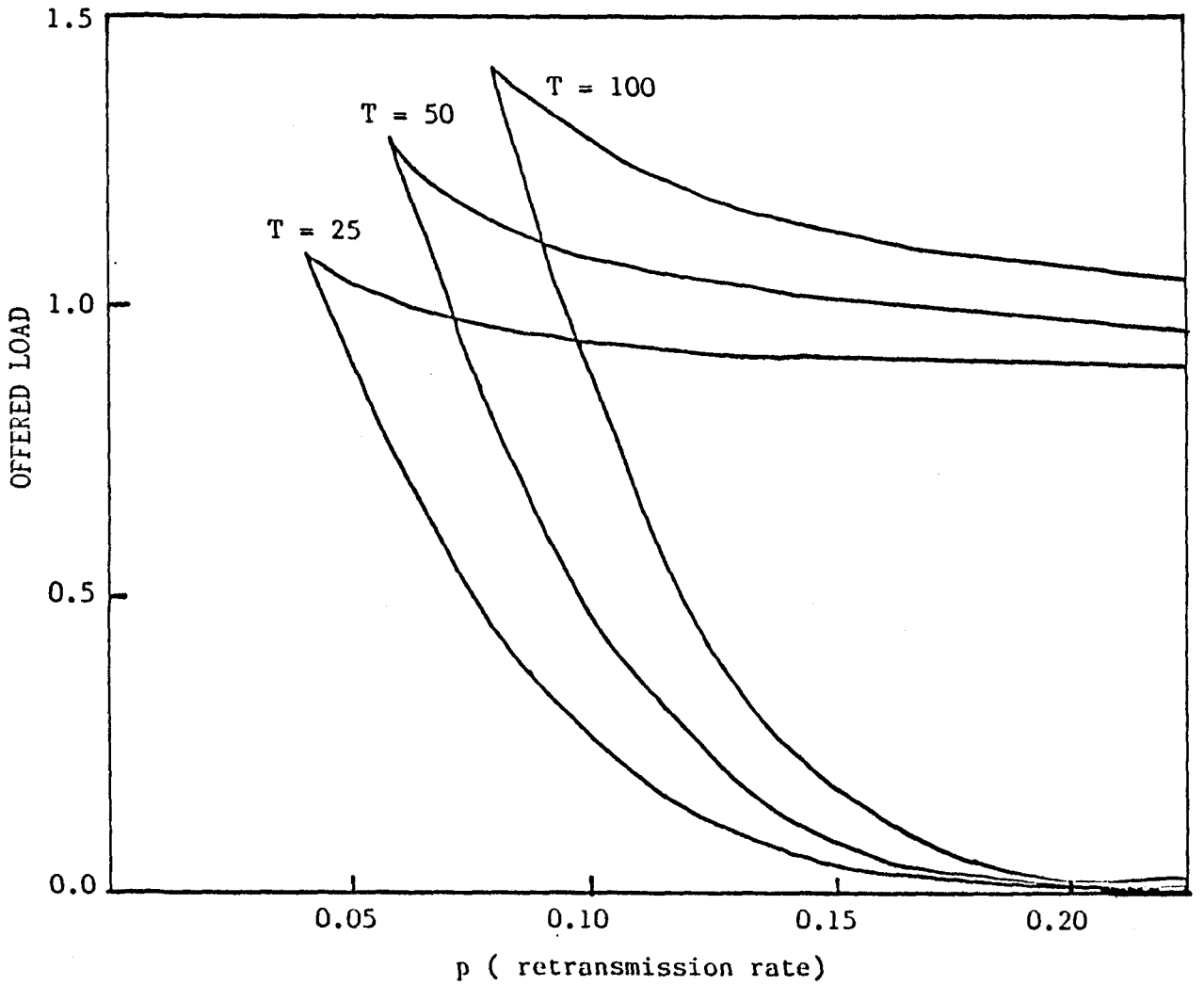


図 3.19: T が変わったときの *Cusp* の変化

4. マルチプログラミングシステムの性能評価

この章で，マルチプログラミングシステムの性能評価を行う．まず，マルチプログラミングシステムのモデル化について述べる．次に，そのモデルを用いて，マルチプログラミングシステムの無限小平均，無限小分散を導出し，ポテンシャル関数を求める．そして，ジョブの発生率，$MMPL$ をコントロールパラメータとしてくさびのカタストロフを応用することによりマルチプログラミングシステムの性能評価を行なう[27,28].

4.1 解析モデル

マルチプログラミングシステムにおいて，メモリ内のジョブ数が $MMPL$ の値に等しい場合，端末から新しく発生したジョブは，メモリを分割使用しているジョブのひとつが実行を終了するまで，CPU の前で待たされることになる．これを考慮したマルチプログラミングシステムのモデルを図 4.1 に示す．

- 端末の総数を M，$MMPL$ を J とする．各端末から発生したジョブは，*CPU-DISK Subsystem* 内のジョブ数が J に等しい場合，*memory queue* で待たされ，J より小さい場合 CPU に移動する．
- CPU で処理されたジョブは確率 p_0 で *SWAP-DISK*，確率 $p_j (j = 1, ..., n$，n は *I/O-DISK* の総数)で *I/O-DISK* j に移動する．ここで，ジョブが *SWAP-DISK* に移動することは，そのジョブが CPU で実行中に *page-fault* が起こったことを意味する．またジョブが *I/O-DISK* に移動することは，そのジョブが CPU で実行中に I/O 処理が発生したことを意味する．
- *SWAP-DISK* で処理されたジョブは CPU に移動する．
- *I/O-DISK* で処理されたジョブは，確率 P_{in} で CPU に，確率 $P_{out} (= 1 - P_{in})$ で *CPU-DISK Subsystem* から退去する．ここで，*CPU-DISK Subsystem* から退去することは，そのジョブの処理が終了したことを意味し，このジョブが退去する際に，*memory queue* で待っているジョブがあれば，*memory queue* の先頭に待っているジョブが CPU に入ることができる．

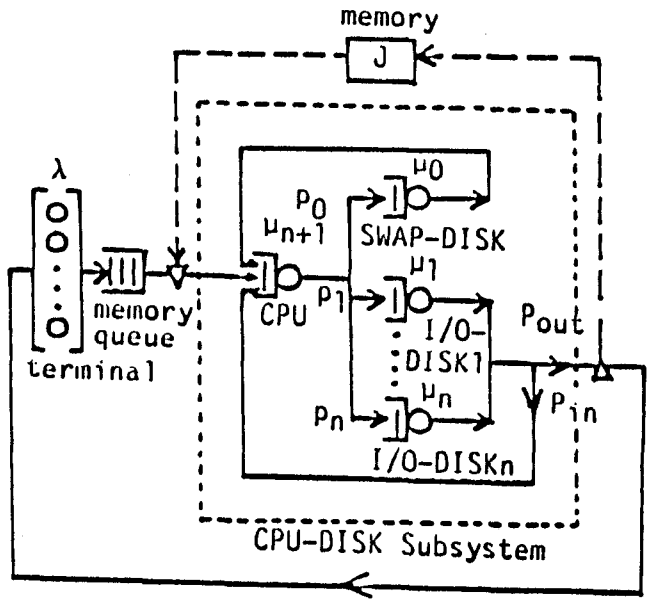


図 4.1: マルチプログラミングシステムの解析モデル

4.2 ポテンシャル関数

マルチプログラミングシステムの状態として，ここでは，*memory queue* で待っているジョブ数と *CPU-DISK Subsystem* に入っているジョブ数の和とし，それを x で表す．前節で述べたモデルの無限小平均$\mu(x;\lambda,J)$，無限小分散$\sigma^2(x;\lambda,J)$ の導出を簡単にするために，下記に述べることを仮定する．

- 各端末を使っているユーザの思考時間（ *Thinking-time*）は，平均$\lambda^{-1}sec$の指数分布とする．但し，ジョブを発生した端末は，そのジョブの実行が終了して端末に戻って来るまで，次のジョブを発生することができないものとする．
- 端末から発生するジョブはすべて同質である．
- その時点に，*CPU-DISK Subsystem* に入っている各々のジョブに対して，メモリは等しく割り当てられる．
- *CPU* の処理規律は *PS(Processor Sharing)* であり，各ジョブは *CPU* において平均$\mu_{n+1}sec$ の指数分布に従う時間，処理を受ける．
- *SWAP-DISK*，および各 *I/O-DISK* の処理規律は *FCFS(First Come First Service)*であり，各ジョブは *SWAP-DISK* および各 *I/O-DISK* において平均$\mu_j sec(j=0,...,n)$ の指数分布に従う時間，処理を受ける．

次に，この仮定をもとに，マルチプログラメングシステムの無限小平均$\mu(x;\lambda,J)$，無限小分散$\sigma^2(x;\lambda,J)$ を求める．

表 4.1 に示すように，端末の数，J に比例してマルコフ解析における状態数は爆発的に増大するので，図 3.6 のモデルを，マルコフ解析を用いて正確に解くことは困難である[29]．よって，ここでは *flow-equivalent* 法[29]を用いて，図 4.1 を近似的に解く．

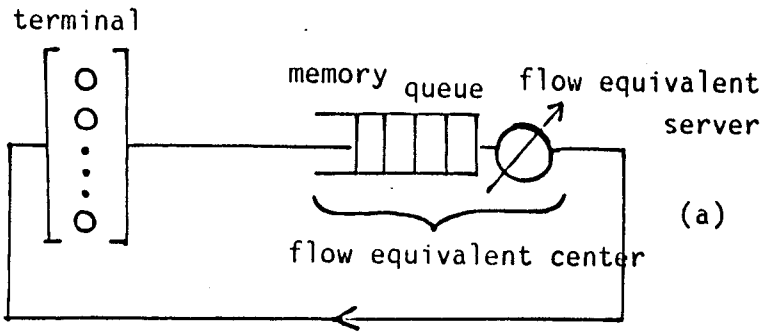
図 4.1 の無限小平均，無限小分散を *flow-equivalent* 法を用いて近似的に解く手順を下記に示す．

(1) 図 4.1 の *CPU-DISK Subsystem* を一つの *server*（以下，この *server* を *flow equivalent server* と呼び，*memory queue* と *flow equivalent server* を併せて *flow equivalent center* と呼ぶ）とする．これにより，図 4.2a が得られる．

(2) 図 4.1 において，*flow equvalent server* とされた *CPU-DISK Subsystem* において，その中のジョブ数が y の時のスループットを求める．つまり，図 4.2b において，$\Lambda_{out}(y)$ を求める．$\Lambda_{out}(y)$ は，*CPU-DISK Subsystem* を系内客数が y である *closed queuing network*[30] と考えて得ることができる．

表 4.1: マルコフ解析における状態数

端末の数 \ MMPL	2	4	6	8	10	12
5	66	196	252	252	252	252
10	141	546	1302	2277	3003	3003
15	216	896	2352	4752	8008	11648
20	271	1246	3402	7227	13013	20748
25	366	1596	4452	9702	18018	29848
30	441	1946	5502	12177	23023	38948
35	516	2296	6552	14652	28028	48048
40	591	2646	7602	17127	33033	57148
45	666	2996	8652	19602	38038	66248
50	741	3346	9702	22077	43043	75348



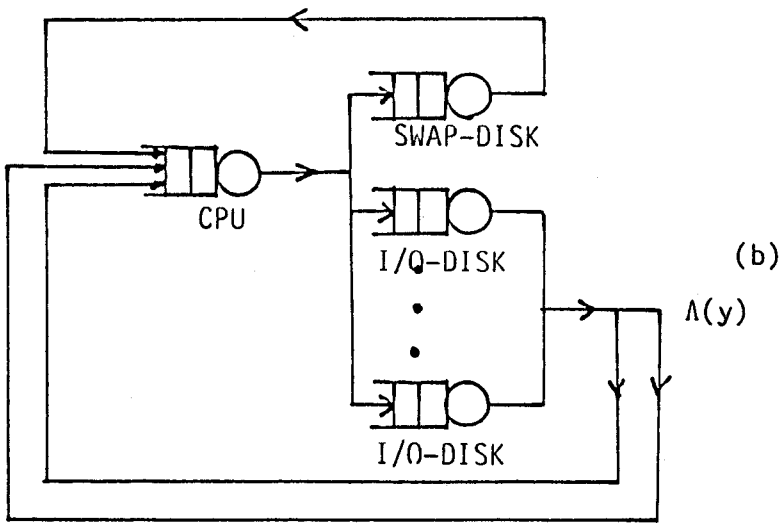


図 4.2: *flow equivalent* 法の近似モデル

(3) (1)で求められた$\Lambda_{out}(y)$ を用いて，図 4.2a のモデルにおいて，*flow equivalent center* の系内ジョブ数が x である時の *flow equivalent server* の処理時間 $C(x)$ を次のようにする．

$$C(x) = \infty \quad , \quad x = 0 \tag{4.1}$$

$$C(x) = \Lambda_{out}(y)^{-1} \quad , \quad x = y = 1, ..., J \tag{4.2}$$

$$C(x) = \Lambda_{out}(J)^{-1} \quad , \quad x > J \tag{4.3}$$

(4) *flow equivalent center* の系内ジョブ数が x の時，*flow equivalent center* へのジョブの到着間隔は平均$(\lambda(M-x))^{-1}$ の指数分布，*flow equivalent center* のジョブの処理は平均 $C(x)$ の指数分布となる．よって，前章における$\mu_a(x)$, $\mu_d(x)$, $\sigma_a^2(x)$ および$\sigma_d^2(x)$ を次のように対応させることによって，

$$\mu_a(x) \implies \lambda(M-x) \tag{4.4}$$

$$\mu_d(x) \implies \Lambda(x) \tag{4.5}$$

$$\sigma_a^2(x) \implies (\lambda(M-x))^{-2} \tag{4.6}$$

$$\sigma_d^2(x) \implies (\Lambda(x))^{-2} \tag{4.7}$$

マルチプログラミングシステムの無限小平均，無限小分散を(4.8)，(4.9)式のように得ることができる．

$$\mu(x;\lambda,J) = \lambda(M-x) - \Lambda(x) \tag{4.8}$$

$$\sigma^2(x;\lambda,J) = \lambda(M-x) + \Lambda(x) \tag{4.9}$$

上記の手順により，マルチプログラミングシステムの無限小平均，無限小分散を求めるためには，図 4.2b における$\Lambda_{out}(y)$ を求めればよい．以下に，$\Lambda_{out}(y)$ を求める方法について述べる．説明の簡単化のために次の記号を定義する．

$R_j(y)$: 系内ジョブ数が y のときの，*queue* j のレスポンスタイム
$L_j(y)$: 系内ジョブ数が y のときの，*queue* j の平均ジョブ数
$\Lambda_j(y)$: 系内ジョブ数が y のときの，*queue* j のスループット

図 4.2b のモデルに対して，*closed queuing network* を解析する手法の一つである *MVA(Mean Value Analysis)* 法[31] を用いる．

図 4.2b のモデルは，先ほどなされた仮定のもとに，積形式解を持ち[30]，積形式解をもつ *closed queuing network* に対して次の定理が成立す[31].

定理1

積形式解を持つ *closed queuing network* において，任意の *queue* に到着した客は，その *closed queuing network* の系内客数が1人少ない場合の定常解を観察する．

この定理より，系内ジョブ数が y である *closed queuing network* において，あるジョブが任意の *queue* j に到着したとき，その *queue* には，平均 $L_j(y-1)$ のジョブ（サービス中のジョブも含めて）が存在している．よって，到着したジョブの *queue* j でのレスポンスタイム $R_j(y)$ は次のようになる．

$$R_j(y) = \frac{1 + L_j(y-1)}{\mu_j}, j = 0, ..., n+1 \tag{4.10}$$

定常状態においては，各 *DISK* において入ってくるフローと出ていくフローは等しいので次の式が成立する．

$$\Lambda_j(y) = p_j \Lambda_{n+1}(y) \tag{4.11}$$

それぞれの *queue* に対して，リトルの公式を適用して，平均系内ジョブ数 $L_j(y)$ は次のようになる．

$$L_j(y) = R_j(y)\Lambda_j(y), j = 0, ..., n+1 \tag{4.12}$$

すべての *queue* の平均系内ジョブ数の和は y に等しいので，次の式が成立する．

$$\sum_{j=0}^{n+1} \Lambda_j(y) = y \tag{4.13}$$

(4.11)式，(4.12)式および(4.13)式より，$\Lambda_{n+1}(y)$ は次のようになる．

$$\Lambda_{n+1}(y) = \frac{y}{R_{n+1}(y) + \sum_{i=0}^{n} p_i R_i(y)} \tag{4.14}$$

いままで求めてきた式を用いて，下記のアルゴリズムにより，$\Lambda_j(y)$ を得ることができる．

(1) $j = 0, ..., n+1$ に対して

$$L_j(0) = 0$$

(2) $y = 1, 2, 3, ...$ に対して

- $j = 0, ..., n+1$ に対して

$$R_j(y) = \frac{1 + L_j(y-1)}{\mu_j}$$

$$\Lambda_{n+1}(y) = \frac{y}{R_{n+1}(y) + \sum_{i=0}^{n} p_i R_i(y)}$$

- $j = 0, ..., n$ に対して

$$\Lambda_j(y) = p_j \Lambda_{n+1}(y)$$

- $j = 0, ..., n+1$ に対して

$$L_j(y) = R_j(y)\Lambda_j(y)$$

よって，$\Lambda_j(y)$ より，$\Lambda_{out}(y)$ は下記のように求まる．

$$\Lambda_{out}(y) = (\sum_{j=1}^{n} \Lambda_j(y)) p_{out} \tag{4.15}$$

式(4.8)，式(4.9)で得られる無限小平均$\mu(x;\lambda,J)$，無限小分散$\sigma^2(x;\lambda,J)$ を用いて，マルチプログラミングシステムのポテンシャル関数は式(3.17)より，次のようになる．

$$V(x;\lambda,J) = -\int_0^x \frac{2\mu(x;\lambda,J)}{\sigma^2(x;\lambda,J)} \tag{4.16}$$

4.3 Cusp

ここでは，マルチプログラミングシステムのポテンシャル関数の形状変化とその *Cusp* について述べる．なお，以下で示す結果は，下記に示すパラメータの値に対するものである．$n = 4$, $M = 100$, $P_{out} = 0.1$, $p_j = (1-p_0)/n (j = 1, \cdots, n)$, $\mu_{n+1} = 0.00005sec$, $\mu_0 = 0.008sec$, $\mu_j = 0.006sec (j = 1, \cdots, n)$.

まず，p_0(*SWAP-DISK* に移動する確率）について簡単に述べる．ある一定量のメモリを持つマルチプログラミングシステムにおいて，メモリに入っているプログラムの数と，*page-fault* が起こる時間間隔との関係は，定性的に図 4.3a のようになる[5]．また，メモリーに入っているプログラムの数と *page-fault* の起こる回数の関係は，定性的に逆の関係にあると考えられる．よって，ここでは p_0 の値を図 4.3b に示すように定める．

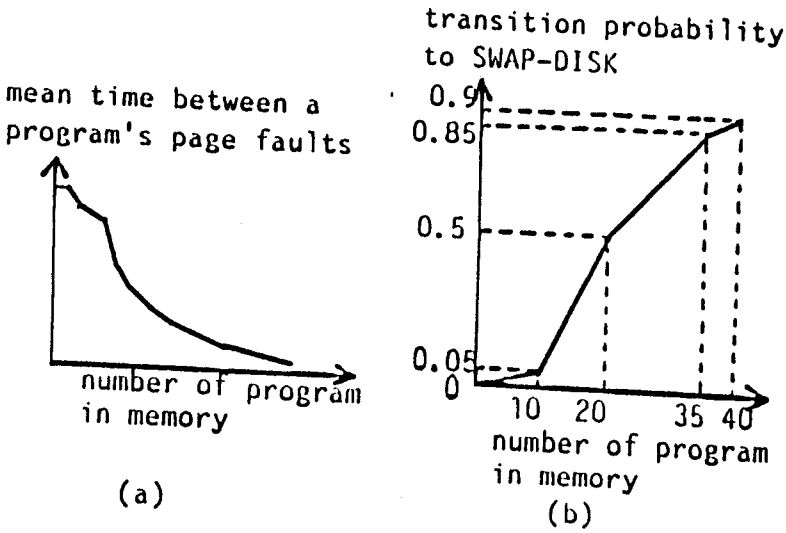


図 4.3: *page-fault* と *SWAP-DISK* にいく確率の関係

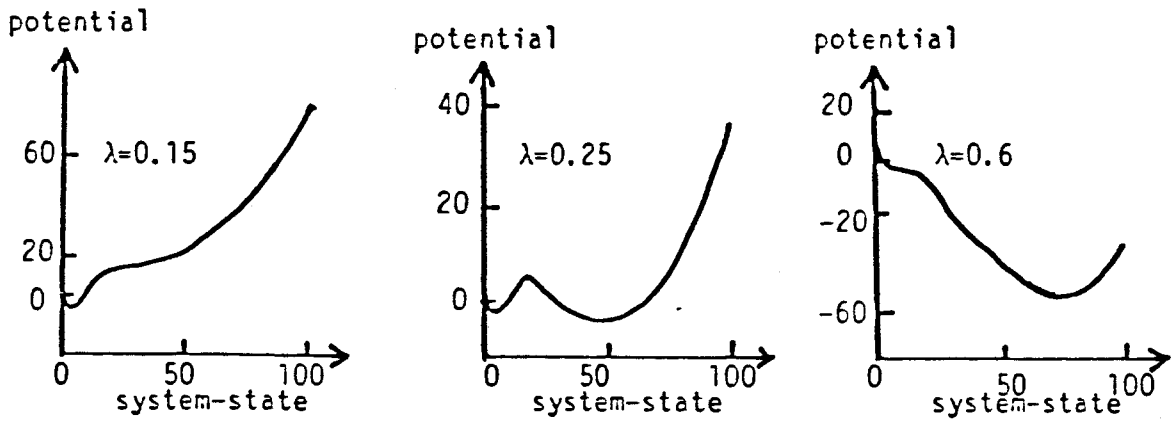


図 4.4: ポテンシャル関数の形状変化（マルチプログラミングシステム）

図 4.4 に，J の値を 20 に固定し，$\lambda = 0.15$，0.25 およびおよび 0.6 に対するポテンシャル関数 $V(x;\lambda,J)$ の形状の変化を示す．図 4.4a，図 4.4b および図 4.4c は，平衡状態の順序がそれぞれ，(安定)，(安定→不安定→安定) および (安定) の例であり，$P(x;a,b)$ の形状の場合分けにおける(1)，(5)，および(1)の場合の形状に対応する．また，$P(x;a,b)$ の形状の(2)，(3)および(4)の場合分けに対応する $V(x;\lambda,J)$ の形状は (λ,J) の値を変えることによって示すことができる．このことより，マルチプログラミングシステムのポテンシャル関数の形状は，$P(x;a,b)$ の変化と同じように変化していることがわかる．よって，マルチプログラミングシステムは，コントロールパラメータとして λ，J を持つくさびのカタストロフになる,ということができる．

マルチプログラミングシステムの *Cusp* を図 4.5 に示す．図 4.4，図 4.5 における $(\lambda,J) = (0.15,20)$，$(0.25,20)$ および $(0.6,20)$ はそれぞれ *Cusp* の下側外部の点 (A 点)，*Cusp* の内部の点 (B 点)，および外側外部の点 (C 点) である．このように，マルチプログラミングシステムのポテンシャル関数の形状変化も $P(x;a,b)$ の形状変化と同じである．よって，システムは *Cusp* の内部で双安定モード，外部で単安定モードになっている．

次に，定常状態確率密度関数の形状変化について述べる．*p-persistent CSMA/CD* 方式の場合と同様に，マルチプログラミングシステムの定常状態密度関数は下記の式で与えられる．

$$f(x;\lambda,J) = Cexp[-V(x;\lambda,J) - ln[\sigma^2(x;\lambda,J)]] \tag{4.17}$$

A 点，B 点および C 点での定常状態確率密度関数を図 4.6 に示す．この図より，*p-persistent CSMA/CD* 方式の定常状態確率密度関数と同様に，マルチプログラミングシステムの定常状態密度関数の形状変化とポテンシャル関数の形状変化は *Cusp* に対して同じになることがわかる．

4.4 平均系内ジョブ数

平均系内ジョブ数 $\bar{x}$ は次の式から求められる．

$$\bar{x} = \int_0^M x f(x;\lambda,J)dx \tag{4.18}$$

$J(MMPL)$の値を固定して，λ の値を変化させたときの平均系内ジョブ数の変化を図 4.7 に示す．

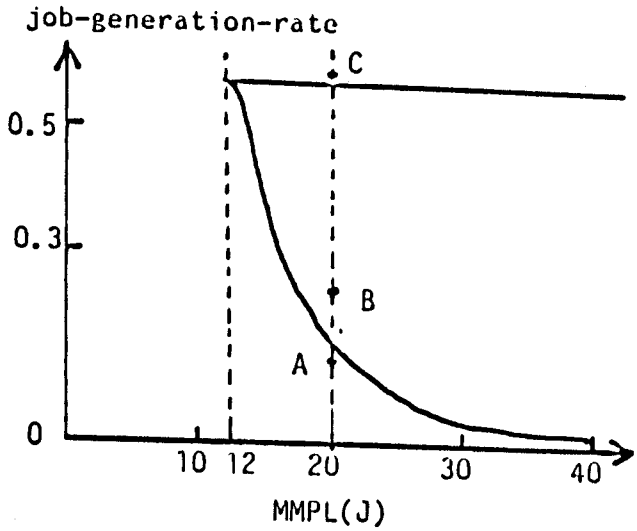


図 4.5: *Cusp*(マルチプログラミングシステム)

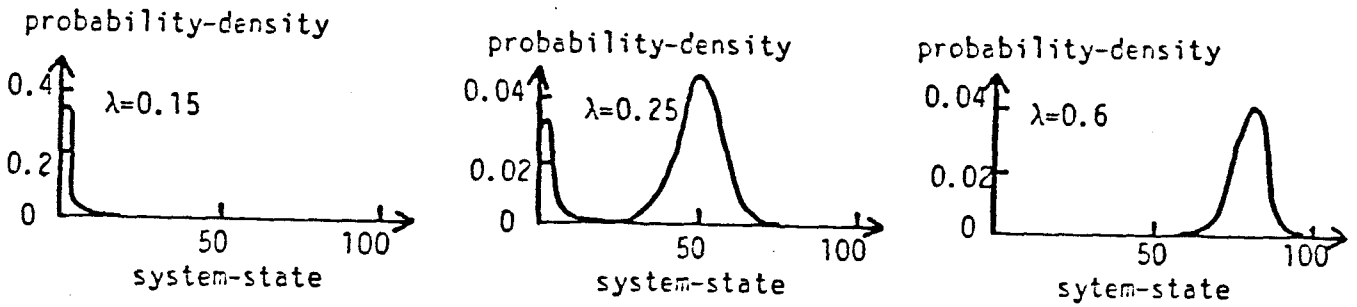


図 4.6: 定常状態密度関数（マルチプログラミングシステム）

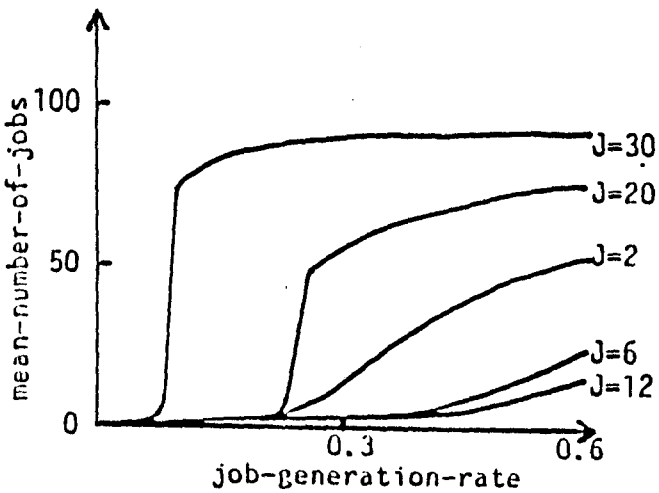


図 4.7: 平均系内ジョブ数（マルチプログラミングシステム）

図 4.5 の(λ, J) 平面において，J が小さいとき$(J \leq 12)$にはλ の値にかかわらず，点(λ, J) は *Cusp* の外部にある．よって，$J \leq 12$ のとき，λ の値を変化させてもシステムは単安定モードのままであり，平均系内ジョブ数はゆるやかに変化している（図 4.7 において，$J = 2, 6$ および 12 の曲線）．ところが，J の値が大きいとき$(J > 12)$にはλ が増大するにつれて，図 4.5 の(λ, J) 平面において，点(λ, J) は *Cusp* の下側から内側を経由して上側に出る．このとき，システムの安定モードは単安定モード→双安定モード→単安定モードと移動し，平均系内ジョブ数は急激に変化する（図 4.7 において，$J = 20$ および 30 の曲線）．このように，コントロールパラメータが *Cusp* を横切るように変化したとき，カタストロフが起こる．

4.5 スループット，レスポンスタイム

スループット$\bar{S}$ は，次の式で求めることができる．

$$\bar{S} = \int_0^M \Lambda(x) f(x; \lambda, J) dx \tag{4.19}$$

J を固定し，λ を変化させたときのスループットの変化を図 4.8 に示す．図 4.8 からわかるように，J の値が小さいとき$(J \leq 12)$には，λ が増加するに従ってスループットもゆるやかに上昇するが，J の値が大きいとき$(J > 12)$は，λ の値を増加させたときに点(λ, J) が *Cusp* の内側に入り，そのときにスループットは急激に落ち込み，一定の値に落ち着く．

レスポンスタイム R は，平均系内ジョブ数$\bar{x}$，スループット$\bar{S}$ を用いて，リトルの公式より次の式で求めることができる．

$$R = \bar{x}/\bar{S} \tag{4.20}$$

J を固定し，λ を変化させたときのレスポンスタイムの変化を図 4.9 に示す．図 4.9 からわかるように，J の値が小さい$(J \leq 12)$ ときはλ の増加に対してレスポンスタイムの増加はわずかであるが，$J = 20$ のときはλ が 0.22 付近から 0.28 付近にかけて，$J = 30$ のときはλ が 0.08 付近から 0.10 付近にかけて，それぞれ急激に上昇している．

図 4.8，図 4.9 からわかるように，ジョブの発生率λ が増加してきたとき，*Cusp* を横切らないような J に対して，スループット，レスポンスタイム共にゆるやかに変化し，*Cusp* を横切るような J に対しては急激に劣化していることがわかる．いいかえれば，システムの安定モードが単安定モード→双安定モード→単安定モードへと

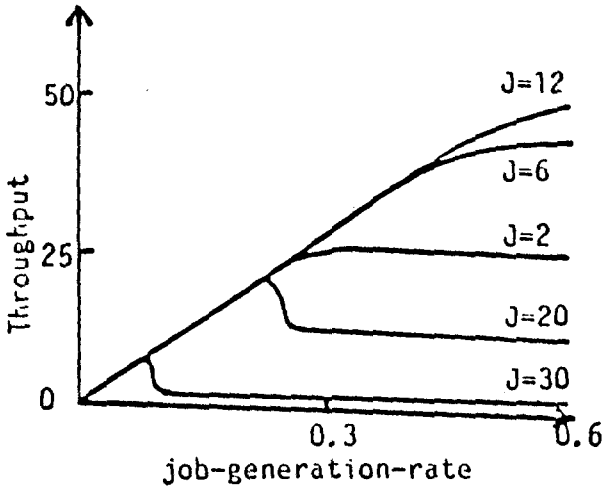


図 4.8: スループット（マルチプログラミングシステム）

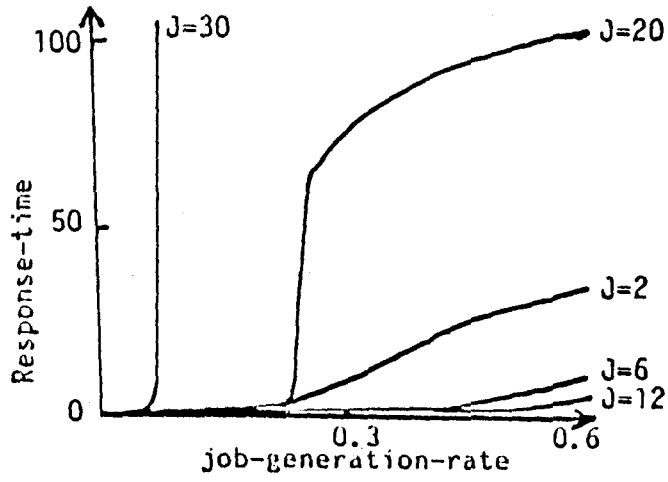


図 4.9: レスポンスタイム（マルチプログラミングシステム）

移り変わっていく過程で，システム性能が急激に劣化することがわかる．また，任意のλの値に対して，$J=12$ のとき（つまり，*Cusp* の頂点の J の値）がスループットは最大，レスポンスタイムは最小になっていることがわかる．よって，これらのことよりマルチプログラミングシステムにおいて，λの変化に対してシステム性能を最適にするためには，J の値を *Cusp* の頂点の値に設定すればよいということがわかる．

5. 回線交換ネットワークの性能評価

この章で，回線交換ネットワークの性能評価をおこなう．まず，対象とする回線交換ネットワークのモデルについて述べる．次に，回線交換ネットワークに対する5つのルーティング方式の概要を述べる．さらに，これらのルーティング方式における*EEBP*を導出し，トラヒックと*EEBP*の関係がカタストロフィー多様体に類似することを示し，*Cusp*を求める．そして，この*Cusp*を用いて，各ルーティング方式の安定性を議論する．最後に，これらから得られた結果をもとにして，トラヒックの変動に対して，ルーティング方式をどのように切り換えればよいか，ということについて議論する[32,33].

5.1 解析モデル

ここでは以下のようなネットワークを対象とする[16,17,18,19].

- ネットワークの形態は完全グラフとする．図5.1にノードの数が5個である時のネットワークの形態を示す．よって，各ノード間には直接つながった*path(direct-path)*が存在し，*direct-path*が*fixed-path*となる．
- 2ホップ以上の*alternate-path*は許さない．例えば，図5.1において，AからBへの*alternate-path*はA→C→B，A→D→BおよびA→E→Bの3つであり，A→E→C→Bのような*alternate-path*は考えない．よって，ノードの数をN，*alternate-path*の数をmとすると，$m = N - 2$の関係がある．また，*alternate-path*に対して，右回りに番号付けを行なう．例えば，図5.1において，A→C→B，A→D→BおよびA→E→Bの*alternate-path*をそれぞれ*1st-alternate-path*，*2nd-alternate-path*および*3rd-alternate-path*と呼ぶ．
- 各ノードには均等にトラヒックがかかるものとする．また，各ノードへの*call*の到着率を平均λのポアソン分布，*call*の*holding-time*を平均$1/\mu$の指数分布とする．よって，各ノードへのトラヒックは$\lambda/\mu(=a_1)$アーランとなる．
- 各ノードを結んでいるリンクの集合を*trunk*と呼ぶ．各*trunk*の大きさ(*trunk-size*)をnとする．つまり，各*trunk*はn本のリンクより構成されているものとする．

また，説明を簡単にするために，図5.1のような完全グラフのネットワークに対して下記の用語を定義する．

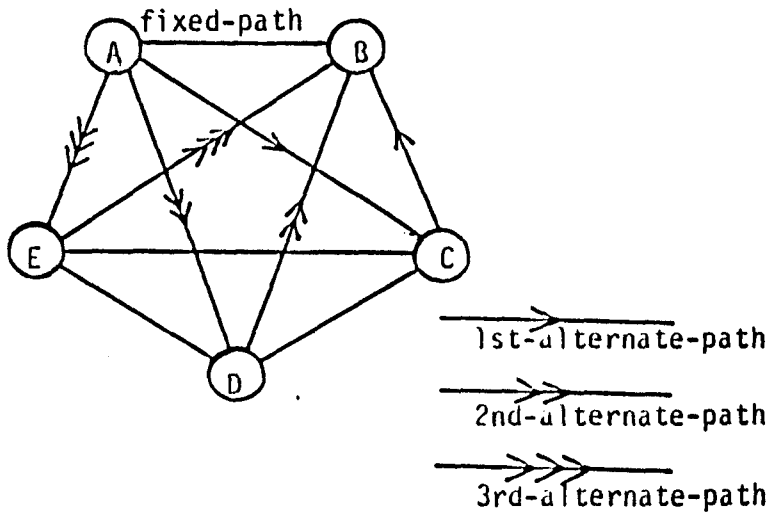


図 5.1: 回線交換ネットワークの解析モデル

trunk-busy: ある *trunk* において，すべてのリンクが使用されていることをいう．逆に，1本のリンクでも空いている場合を *trunk-idle* と呼ぶ．

fixed-path-busy: ある *fixed-path* を形成する *trunk* が *trunk-busy* であることをいう．逆に，*fixed-path* を形成する *trunk* が *trunk-idle* のとき，*fixed-path-idle* と呼ぶ．

alternate-path-busy: ある *alternate-path* を形成する2つの *trunk* のうち，どちらか一方が *trunk-busy* であることをいう．逆に，*alternate-path* を形成する2つの *trunk* がどちらも *trunk-idle* のときを *alternate-path-idle* と呼ぶ．

5.2 回線交換ネットワークおけるルーティング方式

各ルーティング方式およびその *EEBP* の導出について述べる．説明の簡単化のために，呼損系の待ち行列 $M/M/n$ に対する下記のアーランB公式 $EB(n,A)$ を定義する．

$$EB(n,A) \equiv \frac{\frac{A^n}{n!}}{\sum_{i=0}^{n} \frac{A^i}{i!}} \tag{5.1}$$

5.2.1 *NAR* 方式

各ノードに到着した *call* を *fixed-path* のみを用いて伝送する方式である．よって，各ノードに到着した *call* は，*fixed-path-busy* なら呼損となる．この方式は呼損系の待ち行列 $M/M/n$ と考えることができるので，その *EEBP* は下記のようになる．

$$EEBP = EB(n, a_1) \tag{5.2}$$

5.2.2 *AR* 方式

各ノードへ到着した *call* は *fixed-path-idle* ならば *fixed-path* を使用する．もし，*fixed-path-busy* ならば *1st-alternate-path* を使おうと試みる．もし，*1st-alternate-path* が *alternate-path-idle* ならば，*1st-alternate-path* を使用する．もし，*1st-alternate-path* が *alternate-path-busy* ならば，*2nd-alternate-path* を使おうと試みる．このように，*fixed-path-busy* かつすべての *alternate-path* が *alternate-path-busy* ならば，この *call* は呼損となる．

次に，この方式の *EEBP* を導出する．

trunk-busy である確率を y, *trunk-idle* である確率を x とする（ただし, $x = 1-y$). *AR* 方式において, *direct-path*, *alternate-path* にかかるトラヒックは次のように考えることができる.

fixed-path にかかるトラヒック：

call は, まず *direct-path* を使おうとするので, *direct-path* にかかるトラヒックは各ノードへのトラヒック a_1 に等しい.

1st-alternate-path にかかるトラヒック：

1st-alternate-path のトラヒックは, *1st-alternate-path* の 1 番目の *trunk* にかかるトラヒックと 2 番目の *trunk* のかかるトラヒックとの和になる. *fixed-path-busy*(確率 y)で, かつ 2 番目の *trunk* が *trunk-idle*(確率 y)である時のみに, 1 番目の *trunk* にトラヒックがかかる. よって, 1 番目の *trunk* にかかるトラヒックは a_1xy である. 同様に考えて, 2 番目の *trunk* にかかるトラヒックも a_1xy になる. ゆえに, *1st-alternate-path* にかかるトラヒックは $2a_1xy$ で与えれる.

2nd-alternate-path にかかるトラヒック：

1st-alternate-path の場合と同様に, *1st-alternate-path* のトラヒックは, *2nd-alternate-path* の 1 番目の *trunk* にかかるトラヒックと 2 番目の *trunk* のかかるトラヒックとの和になる. *fixed-path-busy*(確率 y), かつ *1st-alternate-path-busy* (確率$(1-x^2)$), かつ 2 番目の *trunk* が *trunk-idle*(確率 y)である時のみに, 1 番目の *trunk* にトラヒックがかかる. よって, 1 番目の *trunk* にかかるトラヒックは $a_1xy(1-x^2)$ である. 同様に考えて, 2 番目の *trunk* にかかるトラヒックも $a_1xy(1-x^2)$ になる. ゆえに, *1st-alternate-path* にかかるトラヒックは $2a_1xy(1-x^2)$ で与えられる.

i 番目の *alternate-path* にかかるトラヒック：

2nd-alternate-path と同様に考えて, i 番目の *alternate-path* にかかるトラヒックは $2a_1xy(1-x^2)^{i-1}$ で与えられる.

AR 方式において, ある *trunk* にかかるトラヒック(A_A)は, その *trunk* が *direct-path* である場合のトラヒックと, *alternate-path* の *trunk* に属する場合のトラヒックの和に等しい. よって, A_A は下記のように与えられる.

$$\begin{aligned} A_A &= a_1 + 2a_1xy + 2a_1xy(1-x^2) \\ &\quad + \cdots + 2a_1xy(1-x^2)^{m-1} \\ &= a_1\left[1 + \frac{2y}{x}(1-(1-x^2)^m)\right] \end{aligned} \tag{5.3}$$

いままでは y を仮定してきたが，いま求めた A_a を用いて y は次のように求まる．

$$y = EB(n, A_A) \tag{5.4}$$

よって，式(5.3)，式(5.4)を連立させることにより，y を求めることができる．

AR 方式において，ある *call* が呼損となるのは，*fixed-path-busy* かつすべての *alternate-path* が *busy* の場合なので，*EEBP* は次の式で与えられる．

$$EEBP = y(1 - x^2)^m \tag{5.5}$$

5.2.3 *AR-EB* 方式

fixed-path，*alternate-path* の選択の順序は *AR* と同じであるが，*1st-alternate-path* が *alternate-path-idle* である場合でも，ある確率$(1-\phi)$ で *1st-alternate-path* を使用することを却下し，*2nd-alternate-path* を選択させ，*2nd-alternate-path* に対しても同様の戦略をとる方式である．よって，$\phi = 0.0$ ならば *AR-EB* は *NAR* と同じであり，$\phi = 1.0$ ならば *AR* と同じである．特に，$0.0 < \phi < 1.0$ のとき，*purely-AR-EB* と呼ぶ．

次に，この方式の *EEBP* について考える．*AR-EB* 方式において，*direct-path*，*alternate-path* にかかるトラヒックは次のように考えることができる．

fixed-path にかかるトラヒック：

fixed-path に対しては *AR* 方式と同じであるから a_1

1st-alternate-path にかかるトラヒック：

fixed-path-busy(確率 y)，かつ 2 番目の *trunk* が *trunk-idle*(確率 y)のときに，確率ϕ で 1 番目の *trunk* にトラヒックがかかる．よって，1 番目の *trunk* にかかるトラヒックは $a_1xy\phi$ である．ゆえに，*1st-alternate-path* にかかるトラヒックは $2a_1xy\phi$ で与えられる．

2nd-alternate-path にかかるトラヒック：

fixed-path-busy(確率 y)，かつ *1st-alternate-path* が使用できない(確率$(1-\phi x^2)$)，かつ 2 番目の *trunk* が *trunk-idle*(確率 y)である時に，確率ϕ で 1 番目の *trunk* にトラヒックがかかる．よって，1 番目の *trunk* にかかるトラヒックは $a_1xy(1-\phi x^2)\phi$ である．ゆえに，*1st-alternate-path* にかかるトラヒックは $2a_1xy(1-\phi x^2)\phi$ で与えられる．

i 番目の *alternate-path* にかかるトラヒック：

2nd-alternate-path と同様に考えて，i 番目の *alternate-path* にかかるトラヒックは $2a_1xy(1-\phi x^2)^{i-1}\phi$ で与えられる．

よって，*AR* 方式と同様に，*trunk* にかかるトラヒック A_E，y，*EEBP* は次のようになる．

$$\begin{aligned} A_E &= a_1 + 2a_1xy\phi + 2a_1xy(1-\phi x^2)\phi \\ &\quad + \cdots + 2a_1xy\phi(1-\phi x^2)^{m-1} \\ &= a_1\left[1+\frac{2y}{x}(1-(1-\phi x^2)^m)\right] \qquad (5.6) \\ y &= EB(n, A_E) \qquad (5.7) \\ EEBP &= y(1-\phi x^2)^m \qquad (5.8) \end{aligned}$$

5.2.4 *AR-TR* 方式

fixed-path，*alternate-path* の選択の順序は *AR* と同じであるが，各 *trunk* において，*alternate-path* が使用することのできるリンクの数に制限を設ける，言いかえれば，*fixed-path* に対してある一定数（この定数を r とする）のリンクを予約する方式である．つまり，*1st-alternate-path* が *alternate-path-idle* である場合でも，*1st-alternate-path* を形成しているどちらかの *trunk* において，空いているリンクの数が r 以下のときには，*1st-alternate-path* を使用することを却下し，*2nd-alternate-path* を選択させ，*2nd-alternate-path* に対しても同様の戦略をとる方式である．よって，$r=0$ ならば *AR-TR* は *AR* と同じであり，$r=n$ ならば *NAR* と同じになる．特に，$0<r<n$ のとき，*purely-AR-TR* と呼ぶ．

次に，*AR-TR* 方式における *EEBP* について考える．x_r を *trunk* において，r 本以上のリンクが *idle* である確率とする．ここでは，ある *trunk* において，空いているリンクの数が r 以下の場合と，r より大きい場合に分けて考える．まず，r 以上の時には，*AR* 方式と同様なので，x_r と式(5.3)を用いて，ある *trunk* にかかるトラヒック $A_{T,1}$ は次のようになる．

$$A_{T,1} = a_1\left[1+\frac{2y}{x_r}(1-(1-x_r^2)^m)\right] \qquad (5.9)$$

r より少ない場合には，その *trunk* は *alternate-path* にならないので，トラヒック $A_{T,2}$ は a_1 に等しい．

次に，*AR-TR* における y，x_rの導出について述べる．いま述べたように，*trunk* にかかるトラヒックは，そのときに空いているリンクの数に依存するので，*AR-TR* 方

式のように，アーランB公式を用いることはできない．しかしながら，状態iとして*busy*であるリンクの数をとれば*AR-TR*方式は，次にような出生率b_i，死滅率d_iをもつ出生死滅過程を考えることができる．

$$\begin{aligned} b_i &= \begin{cases} A_{T,1}\mu_1 & i=0,\cdots,n-r-1 \\ A_{T,2}\mu_1 & i=n-r,\cdots,n-1 \end{cases} \\ d_i &= i\mu, i=1,\cdots,n \end{aligned}$$

よって，*busy*であるリンクの数がi本であるP_iは次のように求まる．

$$P_i = \begin{cases} \frac{A_{T,1}^i}{i!}P_0 & i=0,\cdots,n-r-1 \\ \frac{A_{T,1}^{n-r}A_{T,2}^{i-n+r}}{i!}P_0 & i=n-r,\cdots,n \end{cases} \tag{5.10}$$

P_0は確率の総和が1に等しいので，次の式で与えらる．

$$P_0 = \frac{1}{\sum_{i=0}^{n-r-1}\frac{A_{T,1}^i}{i!} + \sum_{i=n-r}^{n}\frac{A_{T,1}^{n-r}A_{T,2}^{i-n+1}}{i!}}$$

yはすべてのリンクが使われている確率，x_rは*idle*なリンクの数がrより大きい確率であるので，P_iを用いて，それぞれ次のように与えられる．

$$y = P_n \tag{5.11}$$

$$x_r = \sum_{i=0}^{n-r-1} P_i \tag{5.12}$$

よって，*AR-TR*における*EEBP*は次のようになる．

$$EEBP = y(1-x_r^2)^m \tag{5.13}$$

5.2.5 *AR-TR-EB*方式

AR-EB, *AR-TR*両方の戦略を同時に用いる方式である．つまり，ある*alternate-path*を使用できるのは，その*alternate-path*を形成している両方の*trunk*において，空いているリンクの数がrより大きいときであり，その場合でも，確率$(1-\phi)$でその*alternate-path*を使用することを却下される．特に，$0.0 < \phi < 1.0$かつ$0 < r < n$のとき，*purely-AR-TR-EB*と呼ぶ．

ここでも，ある *trunk* において，空いているリンクの数が r 以下の場合と，r より大きい場合に分けて考える．

r より大きい場合にある *trunk* にかかるトラヒックは $A_{C,1}$ は式(5.6)と式(5.9)を同時に考慮した下記の式で表わされる．

$$A_{C,1} = a_1 \left[1 + \frac{2y}{x_r}(1 - (1 - \phi x_r^2)^m)\right] \tag{5.14}$$

r より小さい場合は，*AR-TR* 方式の解析で述べたのと同様に，ある *trunk* にかかるトラヒック $A_{c,2}$ は a_1 に等しい．

この $A_{C,1}$, $A_{C,2}$ を用いて，*AR-TR* と同様の考え方により，P_i, y, x_r は次のように求まる．

$$P_i = \begin{cases} \frac{A_{C,1}^i}{i!} P_0 & i = 0, \cdots, n-r-1 \\ \frac{A_{C,1}^{n-r} A_{C,2}^{i-n+r}}{i!} P_0 & i = n-r, \cdots, n \end{cases} \tag{5.15}$$

$$P_0 = \frac{1}{\sum_{i=0}^{n-r-1} \frac{A_{C,1}^i}{i!} + \sum_{i=n-r}^{n} \frac{A_{C,1}^{n-r} A_{C,2}^{i-n+1}}{i!}}$$

$$y = P_n \tag{5.16}$$

$$x_r = \sum_{i=0}^{n-r-1} P_i \tag{5.17}$$

よって，*AR-TR-EB* における *EEBP* は次のようになる．

$$EEBP = y(1 - \phi x_r^2)^m \tag{5.18}$$

5.3 各ルーティング方式の安定性

この節において，各ルーティング方式の安定性について，議論する．なお，以下の節で示す結果は，$m = 8$（従って，ノードの数 n は 10）に対するものである．

5.3.1 *NAR,AR* の安定性

図 5.2 に，$n = 100$，$n = 50$ および $n = 25$ の時の，*NAR*，*AR* におけるトラヒック（n で正規化）と *EEBP* の関係を示す．この図より，*NAR* はトラヒックの増加に

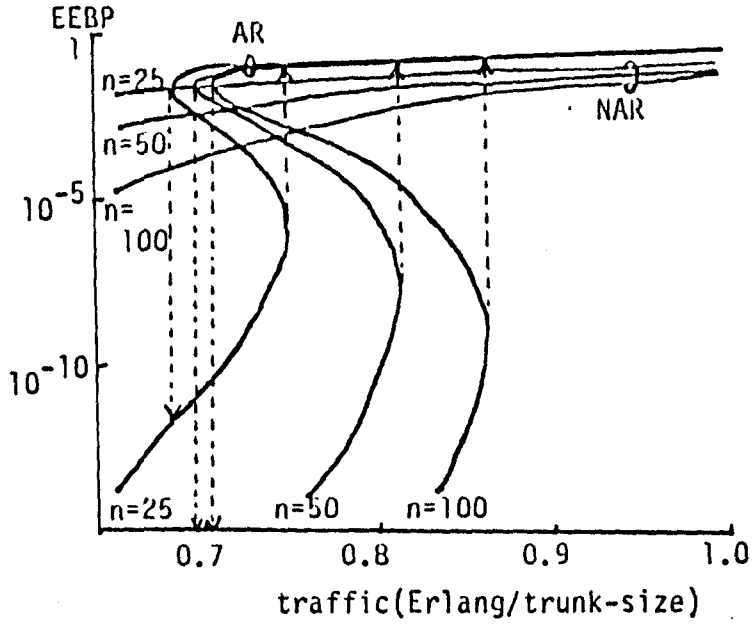


図 5.2: *NAR, AR* における *EEBP* の変化

対して $EEBP$ はゆるやかに増加するが，AR においては，$EEBP$ はトラヒックに対して多価関数になり，トラヒックが増加するにつれて，あるトラヒックの値で $EEBP$ が急激に劣化し，トラヒックが大きい値から減少してくるとき，あるトラヒックの値で $EEBP$ が急激に減少することがわかる．また，トラヒックの小さい時 AR が，トラヒックの大きい時 NAR が優れていることがわかる．

5.3.2 *AR-EB* の安定性

図 5.3 に，$n = 100$ のときの，種々の ϕ の値に対するトラヒックと $EEBP$ の関係を示す．*AR-EB* において，ϕ の値を 1.0 から 0.0 に小さくしていくことは，ルーティング方式が AR から NAR に近ずくことを意味する．よって，ϕ の値が 1.0 に近いときには，$EEBP$ が急激に変化する場合があるが，ϕ の値が 0.0 に近ずくにつれて，$EEBP$ が急激に変化する点がなくなり，$EEBP$ はゆるやかに変化する．また，図 5.3 を 3 次元空間$(a_1, \phi, EEBP)$に拡張すれば，図 5.3 は 2 章で述べたカタストロフィー多様体に類似していることがわかる．よって，それぞれのパラメータを次のように対応させれば，*AR-EB* がくさびのカタストロフになることがわかる．

$[$くさびのカタストロフ$]$		$[AR-EB]$
$normal-parameter: a$	$<------>$	a_1
$splitting-parameter: b$	$<------>$	ϕ
$system-state: x$	$<------>$	$EEBP$

従って，*AR-EB* の *Cusp* を図 5.4 のように得ることができる．$n = 50$，$n = 25$ の場合も同様に *Cusp* を得ることができる．この *Cusp* により，*AR-EB* において，ネットワークトラヒックの増加に対して，トラヒックがどのような値の時に，$EEBP$ の急激な変化が起こり，ネットワークが混雑するか，ということを *Cusp* の上の曲線より知ることができる．また，急激な劣化が起こってしまった時，ネットワークの混雑をなくすためには，トラヒックをどのような値まで下げればよいか，ということも *Cusp* の下の曲線より知ることができる．次に，n の大小に注目して，*Cusp* について考察する．n が小さくなるにつれて，*Cusp* の上の曲線のトラヒックの値（n で正規化）は，小さくなっている．よって，n が小さい方が，トラヒックの増加に対して急激な劣化は起こりやすいことがわかる．しかしながら，n が小さい場合は，急激な劣化が起こってしまった後，ネットワークの混雑をなくすために減らさなければいけないトラヒックは小さくてすむことが *Cusp* の上下の曲線のトラヒックの差からわかる．

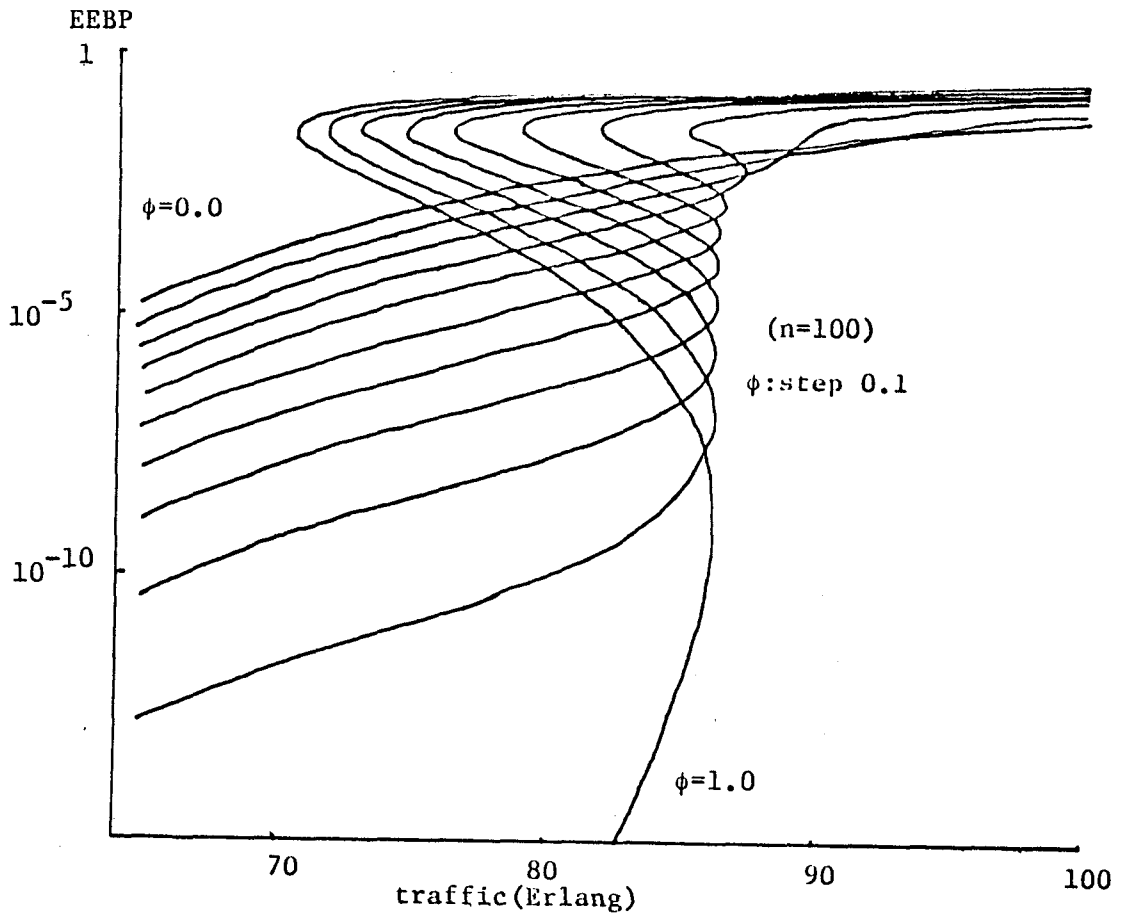


図 5.3: *AR-EB* における *EEBP* の変化

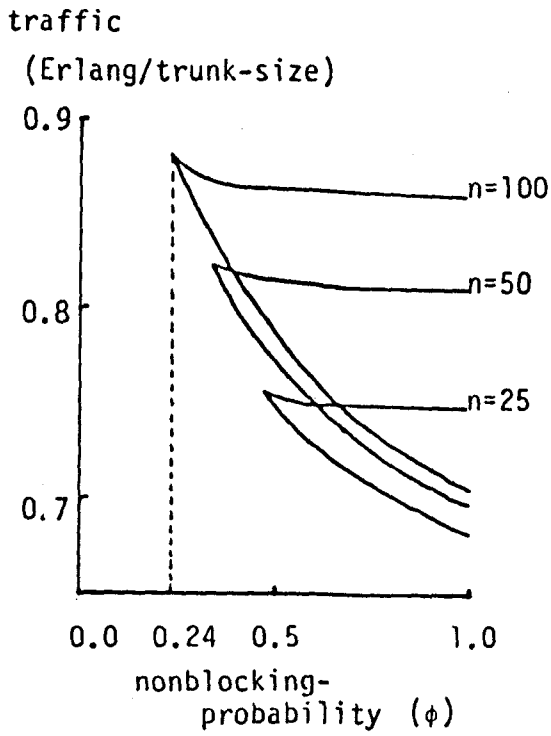


図 5.4: *AR-EB の Cusp*

5.3.3 *AR-TR* の安定性

図 5.5 に，$n = 100$ のときの，r の値を変化させたときのトラヒックと *EEBP* の関係を示す．*AR-TR* において，r の値を 0 から n に大きくしていくことは，ルーティング方式が *AR* から *NAR* に近ずくことを意味する．よって，図 5.5 においても，r が n に近ずくにつれて，変化の様子が *AR* から *NAR* に近ずいている．*AR-TR* においても，*AR-EB* と同様に，トラヒックと *EEBP* の関係を 3 次元空間$(a_1, r, EEBP)$に拡張すれば，カタストロフィー多様体に類似していることがわかる．従ってここでも，それぞれのパラメータを次のように対応させることにより，*AR-TR* がくさびのカタストロフになることがわかる．

[くさびのカタストロフ]		$[AR-TR]$
$normal-parameter: a$	$<------>$	a_1
$splitting-parameter: b$	$<------>$	r
$system-state: x$	$<------>$	$EEBP$

AR-TR の *Cusp* を図 5.6 に示す．n の大小に注目した場合において，*AR-EB* の *Cusp* について述べたことは，*AR-TR* の *Cusp* についてもあてはまる．また，*AR-TR* においては，非常に小さい $r(=0,1,2)$ のときのみ *EEBP* の急激な劣化が起こり，特に，$n = 25$ の時には，*purely-AR-TR* は急激な劣化はない．このように，*AR-TR* は，少数のリンクを予約するだけで，*EEBP* の急激な劣化を防ぐことができる，ということがわかる．

5.3.4 *AR-TR-EB* の安定性

説明を簡単にするために，$n = 100$ の場合だけを考える．また，*AR-TR* において，$r \geq 3$ のときには，急激な劣化が起こらないので，*AR-TR-EB* においては，$r = 0, 1, 2$ のときのみを考える．いろいろな ϕ に対するトラヒックと *EEBP* の関係を図 5.7，図 5.8 に示す．これら 2 つの図もカタストロフィー多様体になっていることがわかる．よって，$r = 0, 1, 2$ のときの *Cusp* が図 5.9 のように得られる．図 5.9 において，$r = 0$ のときの *Cusp* は，$n = 100$ のときの *AR-EB* の *Cusp* と同じものである．このように，*AR-TR-EB* においては，$r = 1$，$r = 2$ のとき，*AR-EB* に比較して，*Cusp* は極端に小さくなっている．

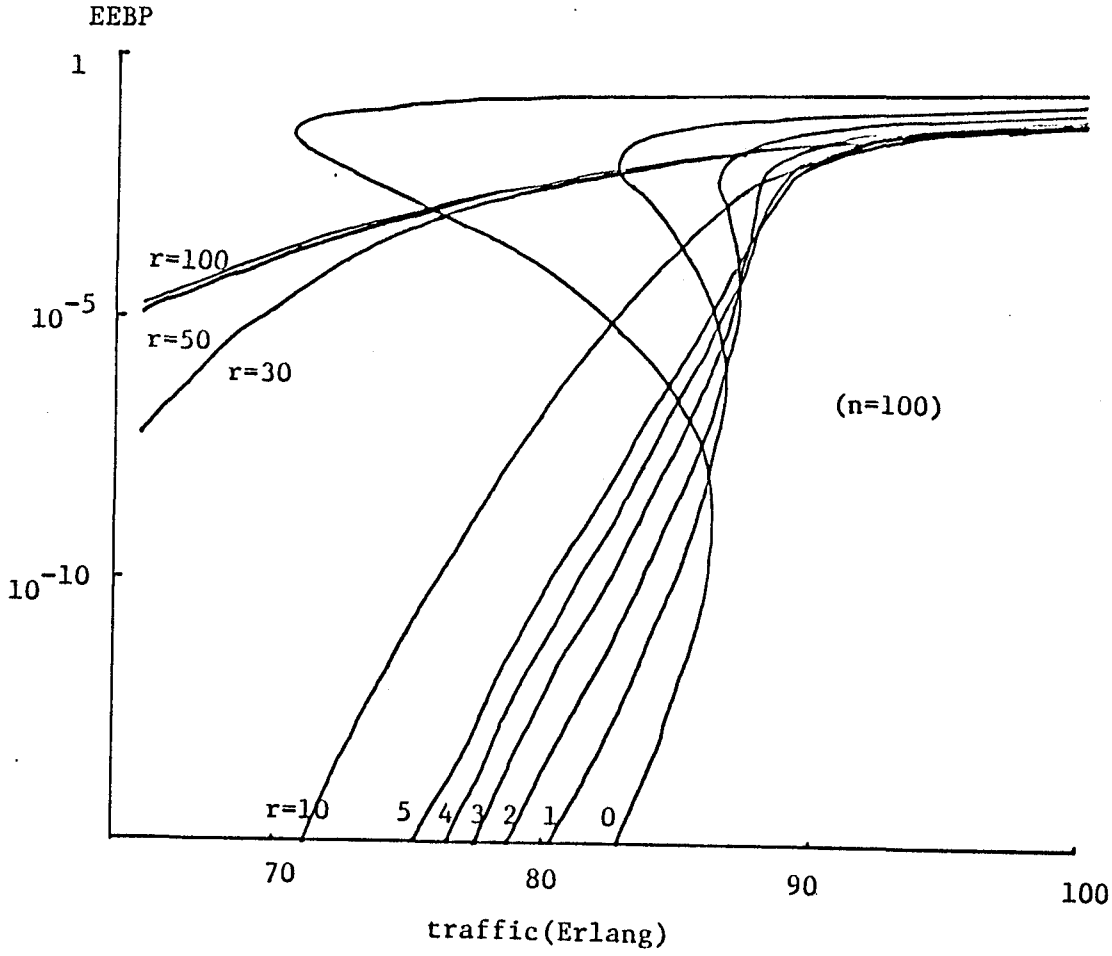


図 5.5: *AR-TR* における *EEBP* の変化

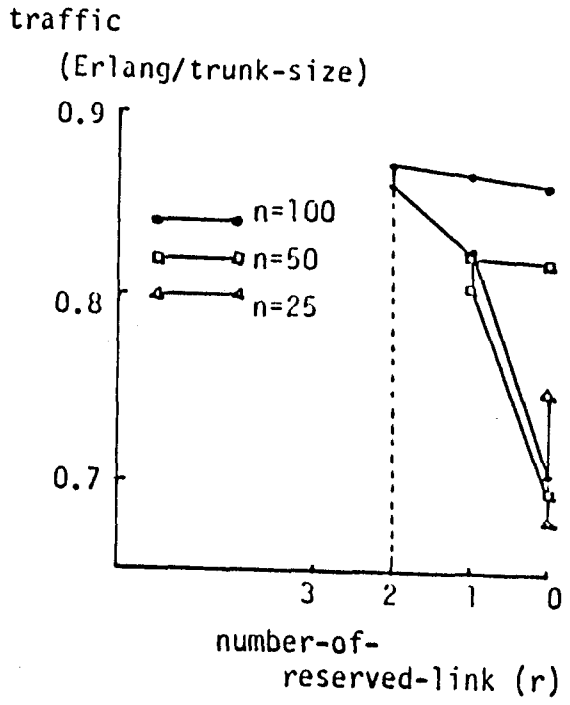


図 5.6: *AR-TR の Cusp*

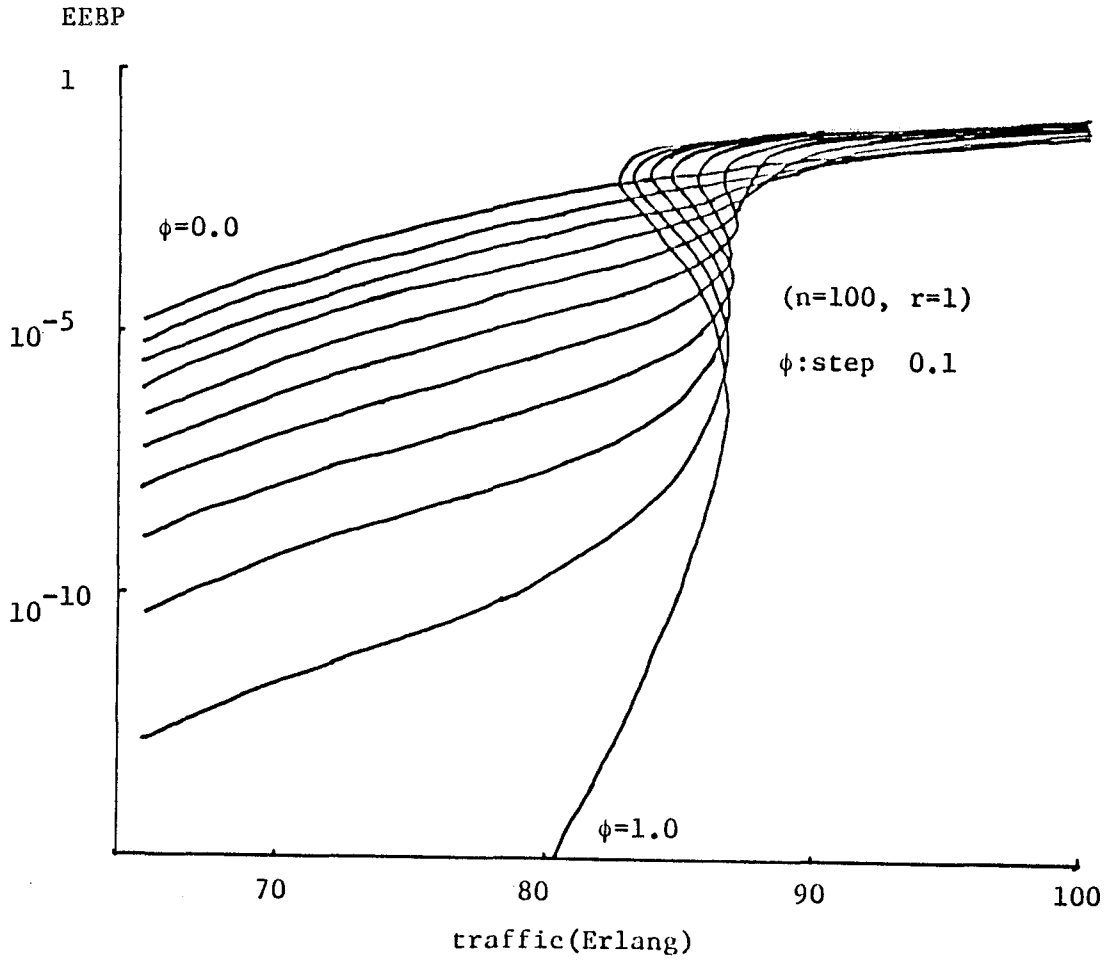


図 5.7: *AR-TR-EB* における *EEBP* の変化($r = 1$)

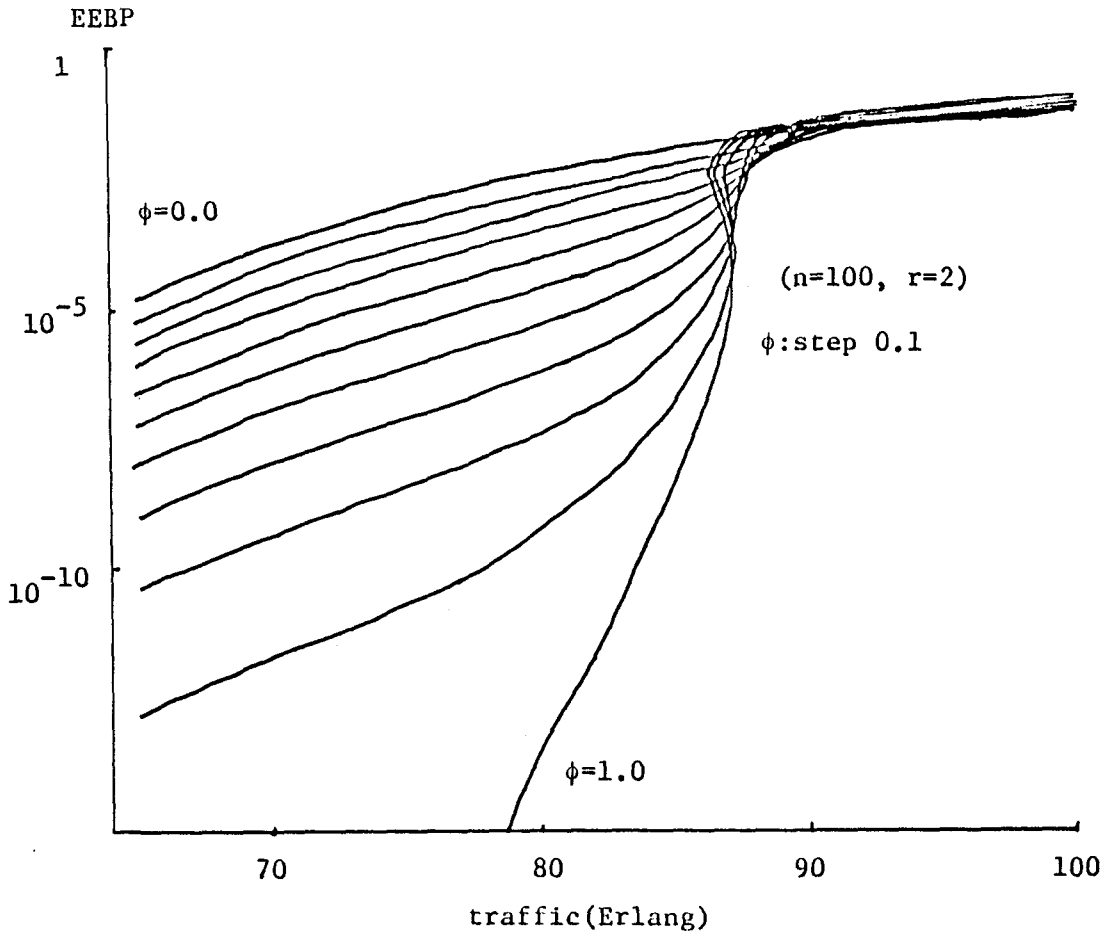


図 5.8: *AR-TR-EB* における *EEBP* の変化($r = 2$)

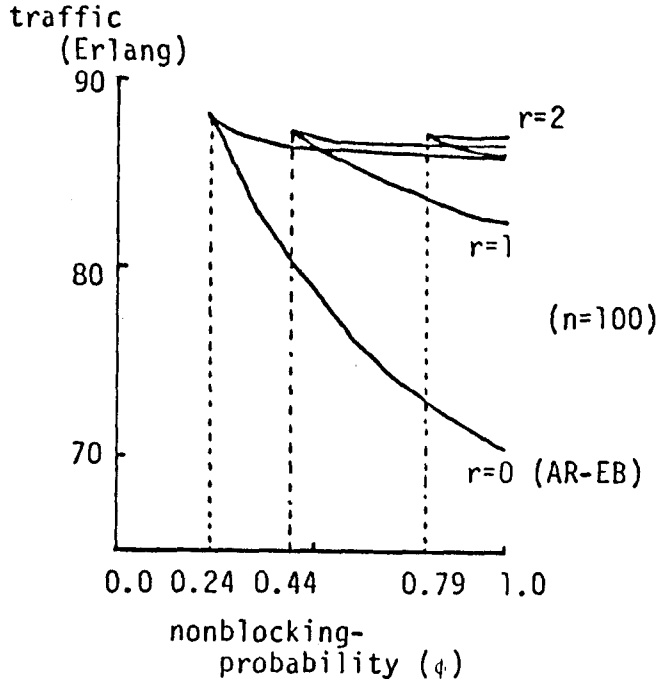


図 5.9: *AR-TR-EB の Cusp*

5.4 安定性の評価

ここでは，それぞれのルーティング式がどれくらい安定か，ということについて議論する．それぞれのルーティング方式の安定性の大小を単純に比較することはできないが，これまで述べてきたように，*Cusp* の *splitting-parameter* の値によって，*normal-parmeter* の値の変化に対して急激な変化が起こるかどうか，ということを決定することができる．よって，ここでは，トラヒックの変化に対して，*EEBP* の急激な変化を防ぐことのできる *splitting-parameter* の範囲の大小に注目し，安定性の一つの評価測度として，安定率を考え，次のように定義する．

定義 6 ：安定率
急激な変化がないような *splitting-parameter* の区間を I, *splitting-parameter* のとりうる区間を I_sとすると，

安定率 $= I/I_s$

また，*AR* においては急激な変化が起こるので安定率は 0.0 , *NAR* においては急激な変化が起こらないので安定率は 1.0 とする．

説明を簡単にするために，$n = 100$ の場合について，安定率を考える．*AR-EB*, *AR-TR* および *AR-TR-EB* に対して，安定率は次のようになる．

AR-EB: $0.24/1.0 = 0.24$
AR-TR: $(100-2)/100 = 0.98$
AR-TR-EB: $r = 0$ の時，*AR-EB* と同じ．
: $r = 1$ の時，$0.44/1.0 = 0.44$
: $r = 2$ の時，$0.79/1.0 = 0.79$
: $r \geq 3$ の時，急激な変化が起こらないので，1.0 とする．

安定率から考えると，*NAR*, *AR-TR* および *AR-TR-EB*($r \geq 3$)は，*AR* および *AR-EB* に比べて，安定性が大きいことがわかる．このように，ある一定数のリンクを予約することは安定性に対して大きく影響している．

5.5 ルーティング方式の切り換え

ネットワークのトラヒックの変化に対して，どのルーティング方式を採用すべきかということは，一般的に難しい問題である．しかしながら，これまで述べてきたことからわかるように，トラヒックが小さい時には AR を，トラヒックが大きい時には NAR を採用した方がよい．この点から考えると，トラヒックの変化に対して，ルーティング方式を AR，NAR，*purely-AR-TR*，*purely-AR-EB* および *purely-AR-TR-EB* のどれかに固定するのはあまりよくないと思われる．よって，ここでは，トラヒックが小さい時には AR を，トラヒックが大きい時には NAR を採用するという立場にたって，トラヒックの増加に対して，*Cusp* を用いてルーティング方式を動的に切り換える方式として，次の 2 つの方式（*deterministic-method*，*control-method*）を考える．

(1) *deterministic-method*: トラヒックが小さいときには，AR を用いる．トラヒックの増加に対して，*Cusp* より *EEBP* が急激に変化するトラヒックの値がわかるので，トラヒックがそのような値になったとき NAR に切り換える．

(2) *control-method*:

(a) *AR-EB* を用いて，トラヒックの増加に対してϕ を次のように動的に変化させる方式である．トラヒックが小さい時には$\phi = 1.0$ とする．トラヒックが増加して，トラヒックが *Cusp* の上の曲線に到達した時，ϕ を *Cusp* の上の曲線に従って減少させる．トラヒックが *Cusp* の頂点に達したときに，ϕ =0.0 とする．つまり，ルーティング方式を AR→*purely-AR-EB* →NAR の順に切り換えることに相当する．

(b) *AR-TR* を用いて，トラヒックの増加に対して r を次のように動的に変化させる方式である．トラヒックが小さい時には $r = 0$ とする．トラヒックが増加して，トラヒックが *Cusp* の上の点に到達した時，r を *Cusp* を用いて，$r = 1$，$r = 2$ と変化させ，トラヒックが *Cusp* の値以上になったとき $r = n$ とする．つまり，ルーティング方式を AR→ *purely-AR-TR*→NAR の順に切り換えることに相当する．*control-method* は，*splitting-parameter* を 2 章で述べたように，*Cusp* を用いて変化させること(*trajectory-3*)に相当する．

$n = 100$ の時の，これら 2 つの方式におけるトラヒックと *EEBP* の関係を図 5.10 に示す．

図 5.2，図 5.3，図 5.5，図 5.7 および図 5.8 と図 5.10 を比較すればわかるように，これらの 2 つの方式が，ルーティング方式を固定した場合に対して劣っているのは，

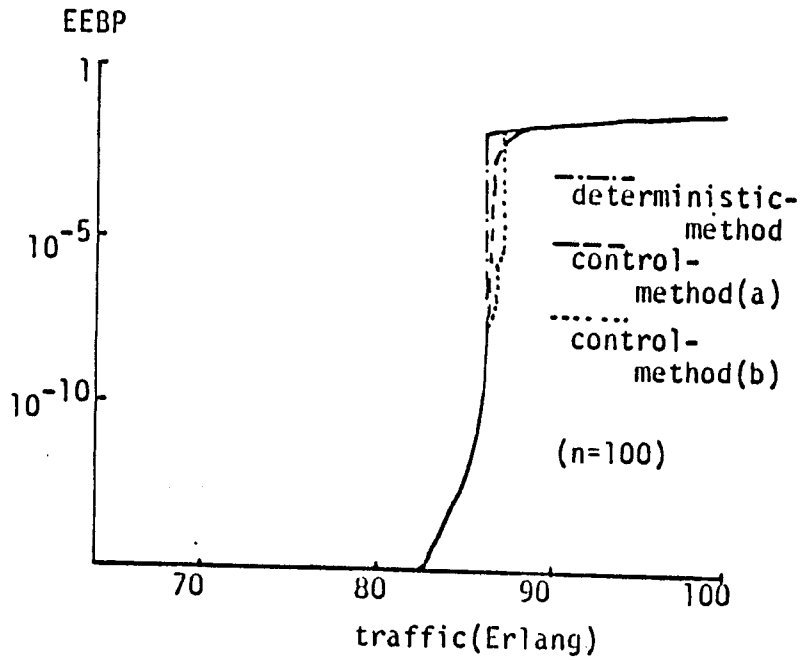


図 5.10: *deterministic-method, control-methor* における *EEBP* の変化

トラヒックの値が86から92の間であり，非常に狭い．よって，これらの方式が有効であることがわかる．

次に，2つの方式について比較する．*deterministic-method* は非常に簡単であるが，*AR* から *NAR* に切り換える時に *EEBP* の急激な劣化が起こる．*control-method* は複雑であり，*deterministic-method* と比較して，それ程，差はない．これは，*Cusp* の上の曲線の傾きが小さいためであり，もし，異なる n，m に対して *Cusp* の上の曲線の傾きが大きくなった場合は，2章で述べたように，*control-method* が有効になると思われる．よって，*Cusp* の上の曲線の傾きが大きい場合は *control-method* を，小さい場合は *deterministic-method* を用いればよいと思われる．

6. 結論

本論文では，双安定特性を有するシステムに対して，(1)システムのモードが双安定モードになるコントロールパラメータの領域（*Cusp*）を求め，(2)コントロールパラメータの変化から引き起こされる安定モードの変化により，システム性能がどのように変化するか，といったことについて考察を行った．

まず，システム状態の変化を解明するための理論であるくさびのカタストロフについて述べた．システムがポテンシャル関数の極小値に対応する状態で安定することを図を用いて説明し，*Cusp* およびカタストロフィー多様体を示した．そして，(1)コントロールパラメータが *Cusp* を横切るようにしたとき，システムの安定状態が急激に変化するが，(2)*Cusp* を横切らない時には安定状態はゆるやかに変化するということを述べた．

次に，くさびのカタストロフをランダムアクセス方式（*slotted-ALOHA*, *p-persistent CSMA/CD*）に応用して，システムの安定モードとシステム性能の関係を明らかにした．システム状態としてシステムに滞留するパケットの個数（*Backlog* モードにあるノードの総数），コントロールパラメータとしてパケットの到着率，再送率をとり，くさびのカタストロフを応用することにより，*Cusp* を求めた．とくに，*p-persistent CSMA/CD* に対しては，*Cusp* とシステム性能の関係を明らかにするために，再送率を固定して，到着率が増加してきた時のシステム性能（スループット，平均パケット伝送遅延）の変化を調べた．その結果，到着率の増加に対して，

- 再送率が小さい時には，システムのモードは単安定のままであり，システム性能はゆるやかに変化するが，
- 再送率が大きいときには，システムの安定モードは，単安定モード→ 双安定モード→ 単安定モードと変化し，それにつれてシステム性能も急激に悪化する

ということを明らかにした．さらに，パケットの到着率が増加した場合，*Cusp* を用いて再送率をコントロールすることによりシステム性能の急激な変化を回避できることを示した．

次に，マルチプログラミングシステムをくさびのカタストロフに応用して，ランダムアクセス方式と同様の考察を行った．システム内に滞留するジョブの総数をシステム状態とし，ジョブの発生率と *MMPL* をコントロールパラメータとして，システムの安定モードとシステム性能を明らかにした．その結果，ランダムアクセス方式と同様に，コントロールパラメータが *Cusp* を横断するときには，システム性能（スルー

プット，レスポンスタイム）が急激に変化することを明らかにした．さらに，マルチプログラミングシステムにおいて，*MMPL* 値を *Cusp* の頂点の *MMPL* の値に設定しておけば，ジョブの発生率がどのようになろうと，システムを最適に保つ（スループットを最大にし，レスポンスタイムを最小にする）ことができるということを明らかにした．

最後に，回線交換ネットワークにおける5つのルーティング方式，つまり，*non-alternate-routing (NAR), alternate-routing (AR), alternate-routing-trunk-reservation (AR-TR), alternate-routing-external-blocking (AR-EB)* および *alternate-routing-trunk-reservation-and-external-blocking (AR-TR-EB)* の安定性を比較・検討し，ネットワークの負荷の増加に対して，ルーティング方式をどのように切り換えればよいか，ということについて述べた．まず，それぞれのルーティング方式の安定性を述べた．特に，*AR-EB*，*AR-TR* および *AR-TR-EB* に対しては，ネットワークの負荷と *EEBP* の関係（カタストロフィー多様体）を示し，負荷の変化に対して *EEBP* が急激に変化する領域 *Cusp* を明らかにした．次に，*Cusp* をもとにして，各ルーティング方式の安定性を比較するために安定率を定義し，*NAR*，*AR-TR* および *AR-TR-EB* は安定率の面からいえば，安定性が大きいということを明らかにした．そして，ネットワークの負荷の増加に対して，*Cusp* を用いてルーティング方式を動的に切り換える方式として，*deterministic-method* と *control-method* を提案し，“*Cusp* の上の曲線の傾きが大きい場合は *control-method* を，小さい場合は *deterministic-method* を用いればよい”ということを明らかにした．

本論文で用いたランダムアクセス方式のモデルにおいては，ノードから発生するパケット長はすべて同じであると仮定した．この仮定は，システムへのトラヒックが1種類であるときには，妥当なものであるが，画像，音声およびテキストなどいわゆるマルチメディアトラヒックが送信されるような状況を考えたとき，パケット長が同じという仮定は非現実的な仮定になってしまう．さらに，音声，画像に対しては，過度の遅延は許されないために，テキストより優先して処理される場合が多い．また，マルチプログラミングシステムのモデルにおいても，すべてのジョブが同質であることを仮定していたが，実際のマルチプログラミングシステムにおいては，いろいろな種類のジョブが混在し，例えばある種のジョブは，他の種類のジョブより優先的に処理されるといったような場合がある．このようにランダムアクセス方式，マルチプログラミングシステムにおいて，システム内に種々のトラヒックが存在するような状況，つまり，より現実的な状況における安定モードとシステム性能の関係の考察が必要であろう．

また，本論文では，回線交換ネットワークの形態が完全グラフ（対称）で，かつ，

各ノードの負荷が均等な場合を扱った．しかしながら，一般的に，ネットワークの形態は非対称で，負荷も均等でない場合が多い．よって，これからの課題として，これら非対称で，不均等なネットワークにおいての各ルーティング方式の安定性の研究を行なう必要があると思われる．

謝辞

本研究をまとめるにあたり，本研究の機会を与えていただき，直接ご指導いただいた大阪大学大型計算機センター宮原秀夫教授に心から感謝致します．

学部，修士課程を通じて御指導，御教授いただいた故高島堅助教授に心から感謝致します．

また，学部，大学院を通じて御指導，御教授いただいた大阪大学基礎工学部情報工学科の故藤澤俊男教授，嵩忠雄教授，鳥居宏次教授，都倉信樹教授，谷口健一教授，並びに大阪大学産業科学研究所の豊田順一教授，角所収教授，北橋忠宏教授に心から感謝致します．

さらに，研究の細部に渡り御討論，ご助言いただいた大阪大学基礎工学部情報工学科の西尾章治郎助教授，日本電気株式会社中央研究所通信研究部の西田竹志氏に心からお礼申し上げます．

最後になりましたが，著者の在学中，御討論いただいた大阪大学基礎工学部情報工学科情報ネットワーク学講座の方々に心から感謝いたします．

参考文献

[1] L. Kleinrock and S. S. Lam, "Packet Switching in a Multiaccess Broadcast Channel: Performance Evaluation," *IEEE Trans. Commun.*, Vol. COM-23, No. 4, pp. 410–423, 1975.

[2] H. Kobayashi, Y. Onozato and D. Huynh, "An Approximate Method for Disign and Analysis of an ALOHA Systems," *IEEE Trans. Commun.*, Vol. COM-25, No. 1, pp. 148–157, 1977.

[3] L. Kleinrock and F. A. Tobagi, "Packet Switching in Radio Channel: Part I- Carrier Sense Multiple Access Modes and Their Throughput-Delay Characteristics," *IEEE Trans. Commun.*, Vol. COM-23, No. 12, pp. 1400–1416, 1975.

[4] F. A. Tobagi and V. B. Hunt, "Performance Analysis of Carrier Sense Multiple Access with Collision Detection," *COMPUTER NETWORKS and ISDN SYSTEMS*, Vol. 4, No. 5, pp. 245–259, 1980.

[5] D. Ferrari, G. Serazzi and A. Zeigner, "Measurement and Tuning of Computer Systems," Prentice-Hall, 1983.

[6] P. Courtois, "Decomposability, Instabilities, and Saturation in Multiprogramming Systems", *Commun. ACM*, Vol. 18, pp. 371–377, 1975.

[7] A. B. Carleial and M. E. Hellman, "Bistable Behavior of ALOHA-Type Systems," *IEEE Trans. Commun.*, Vol. COM-23, No. 4, pp. 401–410, 1975.

[8] 高橋寛子, 松本和良, "ALOHA 方式衛星パケット交換網の解析," 電波研究所季報, Vol. 24, No. 127, pp. 191–203, 1978.

[9] Randolph Nelson, "Stochastic Catastrophe Theory in Computer Performance Modelling," *IBM Reserch Report RC 10406*, 1984.

[10] Randolph Nelson, "The Stochastic Cusp, Swallowtail, and Hyperbolic Umbilic Catastrophes as Manifest in a Simple Communication Model," *Performance 84*, pp. 207–224, 1984.

[11] Randolph Nelson, "Stochastic Catastrophe Theory in Computer Performance Modelling," *J. Assoc. Comput. Mach.*, Vol. 34, No. 3, pp. 661–685, 1987.

[12] Y. Onozato and S. Noguchi, "On the Thrashing Cusp in Slotted Aloha Systems," *IEEE Trans. Commun.*, Vol. COM-33, No. 11, pp. 1171–1182, 1985.

[13] Y. Onozato and S. Noguchi, "A Unified Analysis of Steady State Behavior in Random Access Schemes," *COMPUTER NETWORKS ans ISDN SYSTEMS*,

Vol. 10, No. 2, pp. 111–122, 1985.

[14] Tim Poston and Ian Stewart, "Catastrophe Theory and its Applications," *Pitman*, 1978.

[15] P. T. Saunders, "An Introduction to Catastrophe Theory," *Cambridge University Press*, 1980.

[16] T. G. Yum and M. Schwartz, "Comparison of Routing Procedure for Circuit-Switched Traffic in Nonhierarchical Networks," *IEEE Trans. Commun.*, Vol. COM-35, No. 5, pp. 535–544, 1987.

[17] Y. Nakagome and H. Mori, "Flexible Routing in the Global Communication Network," *Seventh Int. Tele. Cong.*, pp. 426/1–426/8, 1973.

[18] R. S. Krupp, "Stabilization of Alternate Routing," *IEEE Int. Commun. Conf.*, pp. 3I.2.1–3I.2.5, 1982.

[19] J. M. Akinpelu, "The Overload Performance of Engineered Networks with Nonhierarchical and Hierarchical Routing," *Bell Syst. Tech. J.*, Vol. 63, No. 7, pp. 1261–1281, 1984.

[20] T. Yokohira , T. Nishida and H. Miyahara, "Analysis of Dynamic Behavior in p-persistent CSMA/CD Using Cusp Catastrophe," *COMPUTER NETWORKS and ISDN SYSTEMS*, Vol. 12, No. 5, pp. 277–289, 1986.

[21] 横平徳美，西田竹志，宮原秀夫，高島堅助，"カタストロフィー理論を用いた p-persistent CSMA/CD 方式の性能評価," 電子情報通信学会技術研究報告, CS85-59, pp. 7–12, 1985.

[22] L. Kleinrock, "Queueing Systems, Vol.II: Computer Applications," *New York: Wiley*, 1976.

[23] H. Miyahara , T. Matsumoto and K. Takashima, "Transient Analysis of Random Access Method via Diffusion Approximation," *International Conference Modelling Techniques and Tools for Performance Analysis*, pp. 91–108, June 1985.

[24] H. Kobayashi, "Application of the Diffusion Approximation to Queueing Networks I: Equilibrium Queue Distributions," *J. Assoc. Comput. Mach.*, Vol. 21, No. 2, pp. 316–328, 1974.

[25] H. Kobayashi, "Application of the Diffusion Approximation to Queueing Networks II: Nonequilibrium and Applications to Computer Modeling," *J. Assoc. Comput. Mach*, Vol. 21, No. 3, pp. 459–469, 1974.

[26] J. Little, "A Proof of the Queueing Formula $L = \lambda W$" *Oper. Res.*, Vol. 9, pp. 383–387, 1961.

[27] 横平徳美, 宮原秀夫, "マルチプログラミングシステムにおけるカタストロフィー現象の解析," 電子情報通信学会論文誌(D), Vol. J71-D, No. 6, pp. 966–973, 1988.

[28] 横平徳美, 宮原秀夫, "マルチプログラミングシステムにおけるカタストロフィー現象の解析," 電子情報通信学会技術研究報告, SE87-48, pp. 37–42, 1987.

[29] S. Lavenberg, "Computer Performance Modeling Handbook," *Academic Press*, 1983.

[30] F. Baskett , K. M. Chandy , R. R. Muntz and F.G.Palacious, "Open, Closed, and Mixed Networks of Queues with Difference Classes of Customers," *J. Assoc. Comput. Mach.*, Vol. 22, No. 2, pp. 248–260, 1975.

[31] M. Reiser and S. S. Lavenberg, "Mean Value Analysis of Closed Multichain Queueing Networks," *J. Ass. Comput. Mach.*, Vol. 22, pp. 313–312, 1980.

[32] 横平徳美,宮原秀夫, "回線交換ネットワークにおける種々のルーティング方式の安定性の比較," 電子情報通信学会論文誌(B), Vol. J71-B, No. 12, pp. 1411–1418, 1988.

[33] 横平徳美, 宮原秀夫, "回線交換ネットワークにおける種々のルーティング方式の安定性の比較," 電子情報通信学会, 情報ネットワーク・交換ワークショップ, pp. 85–90, 1988.