ソフトウェアの残存エラー数推定モデルの定量的比較

松本　健一　　井上　克郎　　菊野　亨　　鳥居　宏次

（大阪大学　基礎工学部　情報工学科）

要旨：ソフトウェアのテスト段階から収集したデータに基づいて，ソフトウェアの信頼性を定量的に分析，評価するためのモデルが数多く提案されている．本稿では，学生実験から収集したデータを利用して，非同次ポアソン過程（NHPP）の理論を応用したソフトウェア信頼度成長モデル（SRGM）の比較を行う．得られた主な結果を次にまとめる．(1)推定値の誤差評価，及び，信頼度成長曲線の適合度評価に関し，S字型SRGMが指数型SRGMより優れている．(2)習熟S字型SRGMにおけるパラメータrが推定精度に及ぼす影響を明示する．(3)モデルの適用を実時間で実行する時，S字型SRGMが高い推定精度を達成している．(4)超指数型SRGMに対し，推定精度を高める新しい方法を提案する．本稿での結果は，ソフトウェアの開発現場において実際にSRGMを適用する上での基礎を与えると期待される．

1．まえがき

ソフトウェア信頼度成長モデル（SRGM）では，ソフトウェアのテスト段階において発見されるエラーの累積数を時間の関数H(t)として表す．関数H(t)を２次元平面上に描いたときの曲線の形状によって，SRGMは指数型とS字型に分類される．指数型には指数型（Exponential）信頼度成長モデル[1]と超指数型（Hyperexponential）信頼度成長モデル[6,11]，S字型には遅延S字型（Delayed S-shaped）信頼度成長モデル[9]と習熟S字型(Inflection S-shaped)信頼度成長モデル[2,6,8]がある．これらのモデルを比較評価した報告としては文献[4,6,7,10]がある．しかし，通常，１つあるいは複数のプロジェクトに対する１組の収集データに基づいて比較評価が行われる．しかも，データ収集の基本単位が日，あるいは月となっている．

本稿では，大阪大学基礎工学部情報工学科で実施した学生実験から収集したエラーデータに基づいて，上述の４つのSRGMの比較を行う．なお，本比較評価は複数（ここでは５）組の，しかも，分単位の収集データに基づいている．

得られた主な結果を次にまとめる．

(1)推定精度の評価： 初期エラー数の推定値と実測値の間の誤差評価，および，信頼度成長曲線の適合度評価に関しては，S字型モデルが指数型モデルより優れていることが分かった．

(2)rの推定： 習熟S字型モデルにおけるパラメータrと初期エラー数の推定精度の関係について調べた．rを0に近い値（例えば，r＝0.1）にすれば，かなりの良い精度が実現できることが確認できた．

(3)テストの進捗に伴う評価： テスト中の各時点で，利用可能なデータのみに限定してモデルを適用した場合の推定精度の振る舞いについて評価した．遅延S字型モデルによる推定だけが比較的安定していることが分かった．

(4)原点ずらし： 超指数型信頼度成長モデルに対し，推定の精度を高める方法を提案した．この方法はプログラムのクラスタ毎にテスト開始時刻が異なること，及び，データ収集の基本単位が小さいことに注目したものである．

なお，得られた結果をソフトウェア開発の現場への応用についても議論する．

2．準備

A．ソフトウェア信頼度成長モデル

プログラムテストの実行で，プログラム中のエラーは発見され，取り除かれる．今，エラーを取り除く過程で新たなエラーがプログラム中に作り込まれないと仮定すると，プログラムテストの進行に伴ってプログラムの信頼度は高くなる．

プログラムの信頼度に関するこの現象は，い

わゆるソフトウェア信頼度成長モデル（SRGM）を利用すると説明がつく．SRGMでは，テスト時間と発見されるエラーの累積数の間の関係を表現する信頼度成長関数H(t)（以降では，信頼度成長曲線とも呼ぶ）を用いてソフトウェアの信頼度を定義する．

SRGMは信頼度成長曲線の形状によって，指数型SRGMとS字型SRGMに分類される（図1参照）．代表的な指数型SRGMには，GoelとOkumoto[1]が提案した E(Exponential)モデルM_{GO}，大場ら[6,11]が提案した HE(Hyperexponential)モデルM_{HE}などがある．一方，S字型SRGMには，山田ら[9]が提案した DS(Delayed S-shaped)モデルM_{DS}，大場[2,6,8]が提案したIS(Inflection S-shaped)モデルM_{IS}などがある．

各モデルにおける信頼度成長曲線H(t)の定義式を表1にまとめる．表1中で，tはテスト中の時刻，Nはテスト開始時にプログラム中に存在したエラーの総数（初期エラー数），ϕはエラーの発見率，rはエラーの総数に対する発見可能なエラーの総数の割合を表す．なお，HEモデルM_{HE}は幾つかのクラスタ（機能的にまとまったモジュール群）から構成されるプログラムの解析に用いられる．そこで，N_i，ϕ_iは，それぞれ，クラスタiに関するN，ϕである．

本稿では，5人の学生によるコンパイラ作成の学生実験から収集したデータを用いて，上述の4つのSRGM，M_{GO}，M_{HE}，M_{DS}，M_{IS}の比較を行う．

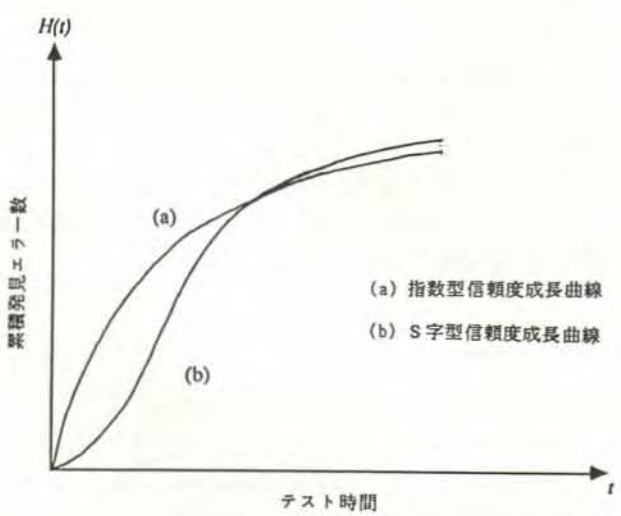


図1　ソフトウェア信頼度成長曲線

B．パラメータ

表1に示す様に，関数H(t)を定めるにはパラメータN，ϕ，(M_{IS}に対しては)rを求める必要がある．なお，Nとϕは実験データに基づいて最ゆう推定法によって求めた(詳細は文献[6]の付録A-Cを参照されたい)．

C．評価基準

比較の基準として次の2つを考える．

(1) 初期エラー数に対する推定の精度 A

$$A = | N_0 - N | / N_0$$

ここで，N_0，Nはそれぞれ初期エラー数の実測値，推定値とする．

(2) Kolmogorov-Smirnov検定による信頼度成長曲線H(t)の適合度 κ

D．実験データ

情報工学専攻の約40人の学生を対象に実験データを収集した．ここでは，予め定めた期間内に実験を終了した5人の学生（x125，x132，x133，x134，x136）に関する実験データを利用する．

各学生は言語Pascalの部分集合に対して，言語Cを用いてコンパイラを作成した．どの学生もコンパイラ作成に関する講義を聴講しているが，実際の作成経験はなかった．作成されたプ

表1　各モデルにおける信頼度成長曲線H(t)の定義

モデル		信頼度成長曲線H(t)
指数型モデル	M_{GO}	$H(t) = N[1-\exp(-\phi t)]$
	M_{HE}	$H(t) = \sum_{i=1}^{n} N_i[1-\exp(-\phi_i t)]$
S字型モデル	M_{DS}	$H(t) = N[1-(1+\phi t)\exp(-\phi t)]$
	M_{IS}	$H(t) = N\dfrac{1-\exp(-\phi t)}{1+\frac{1-r}{r}\exp(-\phi t)}$

表2　実験データ

n_i	t_i (min.)				
	x125	x132	x133	x134	x136
1	36	145	59	234	103
2	40	189	73	371	176
3	45	356	101	388	299
4	52	437	154	395	405
5	54	437	157	467	434
6	102	531	173	521	434
7	108	621	199	521	450
8	108	634	210	574	483
9	172	642	352	609	491
10	197	867	418	609	591
11	200	909	431	628	894
12	242	913	581	665	973
13	326	969	597	797	973
14	491	975	630	802	1009
15	514	1154	641	862	1070
16	514	1216	646	1074	1096
17	514	1237	648	1174	1157
18	514	1454	679	1328	1158
19	514	1527	796	1349	1346
20	537	1538	801	1409	1346
21	580	1538	866	1424	1486
22	677	1573		1435	1525
23	678	1959		2295	1960
24	678	2241		2319	1967
25	732			3119	2065
26	779			3803	2133
27	794			3803	2530
N_r	5	1	2	1	1

ログラムの大きさは行数にして約1000行であった．プログラム作成とプログラムテストは同一の学生が行った．なお，最終的なエラー数を求めるため，実験終了後に筆者らがプログラムテストを更に行った．

表2に5人の学生から収集した実験データを示す．t_iは観測時刻，n_iは時刻t_iまでに発見されたエラーの累積数を表す．N_rは実験終了後にプログラム中に残っていたエラーの総数である．なお，本実験ではt_iは累積端末使用時間で表す．n_iの値はプログラムテキストの更新記録を分析して求めた．N_rの値は学生実験終了後に筆者らが行ったプログラムテストによって求めた．

表2の実験データの主な特徴を次に示す．

C1　5組の実験データが利用できる．

C2　エラー発見時刻が分単位で求められている．これは，ソフトウェアの信頼度評価の通常の実験データに比べて詳細である．

C3　プログラムテストの期間が短い．（高々，3803分＝2.5日の長さである．）

C4　プログラムの大きさが比較的小さい．（約1000行）

C5　プログラムが2種類のクラスタ（構文解析部とコード生成部）から構成されている．この構成に対応させて表2を分けると，表3が求まる．

表3　クラスタ毎の実験データ

(a)構文解析部(Parser)

n_i	t_i (min.)				
	x125	x132	x133	x134	x136
1	40	145	59	234	103
2	52	189	73	388	299
3	54	356	101	1174	491
4	102	531	154	1328	591
5	108	621	157	1349	894
6	108	642	199	3803	973
7	172	1154	352	3803	1009
8	197	1216	431		1070
9	200	1237	581		1486
10	242	1527	630		1960
11	326	1538	641		1967
12	491	1573	646		2065
13	514	2241	866		2133
14	514				2530
15	514				
16	514				
17	537				
18	580				
19	677				
20	678				
21	678				
22	732				
23	779				
24	794				

(b)コード生成部(Generator)

n_i	t_i (min.)				
	x125	x132	x133	x134	x136
1	36	437	173	371	176
2	45	437	210	395	405
3	514	634	418	467	434
4		867	597	521	434
5		909	648	521	450
6		913	679	574	483
7		969	796	609	973
8		975	801	609	1096
9		1454		628	1157
10		1538		665	1158
11		1959		797	1346
12				802	1346
13				862	1525
14				1074	
15				1409	
16				1424	
17				1435	
18				2295	
19				2319	
20				3119	

3. 推定精度の評価

ここでは，指数型モデルM_{GO}とS字型モデルM_{DS}，M_{IS}の間の比較を行う．

3.1 パラメータr

パラメータrはエラーの総数に対する発見可能なエラーの総数の割合である[6]．rの値の決定は，一般的に，非常に困難である．今回の実験では，作成するプログラムが構文解析部とコード生成部の2つの部分から構成されている事実に注目した．しかも，構文解析部の後にコード生成部が位置している．そこで，構文解析部のエラーの総数が，近似的に，発見可能なエラーの総数に等しいと仮定した．よって，エラーの総数で構文解析部のエラーの総数を割った値をrと定める．

3.2 比較結果

モデルM_{GO}，M_{DS}，M_{IS}を5人の学生から収集した実験データに適用した結果を表4に示す．$\bar{A}$は5人の学生に対するAの平均を示す．pはKolmogorov-Smirnov検定でH(t)が実測値と適合していると判断された割合を示す．

表4中の$\bar{A}$を見ると，M_{DS}がM_{GO}とM_{IS}より優れていることが分かる．しかも，M_{DS}における$\bar{A}$の値は10%以下で，精度が非常に高い．

次に，曲線H(t)の適合度に関してはS字型モデルのM_{DS}が$p=5/5$，M_{IS}が$p=3/5$である．これに対し指数型モデルM_{GO}は$p=1/5$である．従って，S字型モデルが指数型モデルより優れていることが分かる．

以上の結果から見ると，S字型モデルが指数型モデルより優れていると考えられる．学生x132に対する各モデルの信頼度成長曲線H(t)を図2に示す．

4. モデルM_{IS}におけるrの推定

4.1 実験の背景

S字型モデルM_{IS}においては，関数H(t)の決定にパラメータrの値が前提となる．3.1でも説明した様に，rの値の決定は，一般的に，非常に困難である．実用的には，過去の経験やデータに基づいてrを決定すべきである．

3.1の実験では，作成するプログラムの構造に注目してrを定義する試みを行っている．一方，プログラムテストの現場では，2，3の値を仮に代入してみて，最も精度の良いものを選ぶ方法がとられている．ここでは，パラメータrを0～1の範囲で0.05毎に変化させて，初期エラー数の推定精度の変動を評価してみた．

4.2 実験結果

主な結果を図3と表5に示す．図3の横軸はrの値（r＝0.01，0.05，0.10，0.15，…，0.95，1.00）を，縦軸は推定精度＝$(N-N_0)/N_0\times100$(%)を表す．図3(b)の太い実線は，N_0の値が∞となった区間を示している．表5における最適なrの値は，rを0.01毎に変化させて$N=N_0$となる時点を求めることで定めた．

5人の学生の内で，x132，x134，x136の3人については図3(a)に示す様に，右上がりの直線となった．これに対し，x125とx133の2人については図3(b)の様に，rが0の近傍では右上がりの直線となっているが，rの値が増大すると$N=\infty$となる．

5. テストの進捗に伴う評価

5.1 実験の背景

パラメータN，ϕ，rの計算を実時間で，すなわちテストの進捗状況に応じて実行することについて議論する．従って，3.では表2の全ての実験データを利用できたのに対し，ここでの評価では最初のデータ（時刻t_iではn_i個のデータ）だけが利用可能となる．

こうした評価データの蓄積により，例えば，(1)より現実的な環境下でのモデルの比較評価，(2)テスト段階での終了時期の決定，(3)テストを実施するスタッフ構成の改善，などが可能になる．

5.2 比較結果

学生x132に対する比較結果を図4に示す．他の4人（x125，x133，x134，x136）についてもほぼ同様の傾向が観察できる．

図4より，グラフは全体的に右上がりである．即ち，利用可能なデータ数が増大するに伴い，Nの値も大きくなる傾向にある．しかし，Nの値は突発的に∞になってしまう．全体的にはS字型モデルM_{DS}が比較的安定している．

利用可能なデータ数が15の場合，S字型モデルM_{DS}とM_{IS}の両方が，ともに良い推定精度を実現している．その状況を直観的に示すため，学生x132に対するM_{DS}とM_{IS}の信頼度成長曲線H(t)を図5に示す．なお，M_{IS}の適用では4.2で求めた最適なrの値を利用している．

表4　モデル適用結果

		x125	x132	x133	x134	x136
初期エラー数　N_0		32	25	23	28	28
M_{GO}	N ϕ ($\times 10^{-3}$)	64.1 0.69	40.2 0.41	111.7 0.24	28.9 0.71	37.6 0.50
$\bar{A}$-116.8% p-1/5	A κ	100.4% ×	61.0% ×	385.4% ×	3.3% ○	34.1% ×
M_{DS}	N ϕ ($\times 10^{-3}$)	31.6 4.29	26.7 1.74	26.4 3.41	27.3 1.72	28.9 1.74
$\bar{A}$-8.6% p-5/5	A κ	1.3% ○	6.6% ○	29.4% ○	2.5% ○	3.3% ○
M_{IS}	N ϕ ($\times 10^{-3}$) r	61.9 0.79 0.89	29.8 0.96 0.54	76.9 0.55 0.62	27.3 1.59 0.26	31.2 1.03 0.52
$\bar{A}$-72.2% p-3/5	A κ	93.4% ×	19.2% ○	234.3% ×	2.7% ○	11.3% ○

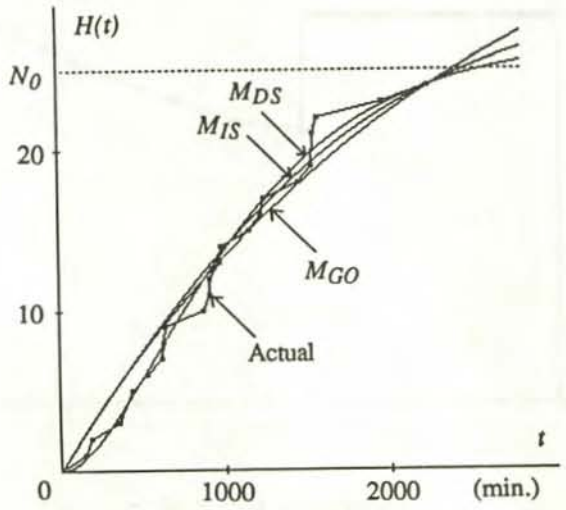
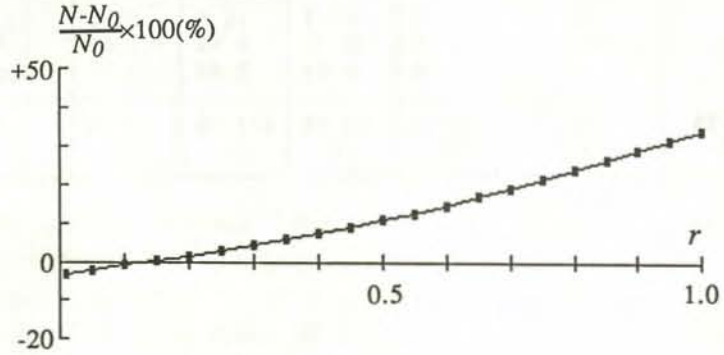


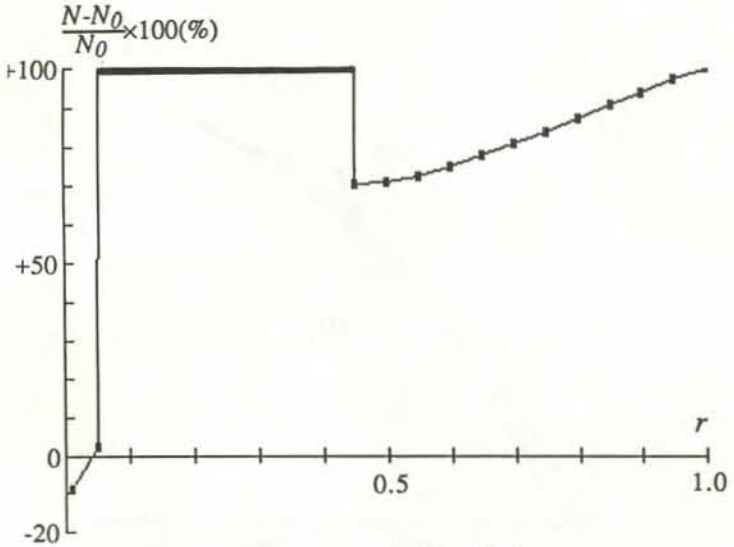
図2　信頼度成長曲線の比較（x132）

表５　初期エラー数Ｎ$_0$の推定

		x125	x132	x133	x134	x136
初期エラー数　Ｎ$_0$		32	25	23	28	28
3.1で定めたｒを用いた場合	Ｎ	61.9	29.8	76.9	27.3	31.2
	ｒ	0.89	0.54	0.62	0.26	0.52
最適なｒを用いた場合	Ｎ	31.8	25.0	23.2	28.0	28.0
	ｒ	0.04	0.11	0.02	0.66	0.14



(a)推定精度が安定している場合



(b)Ｎ＝∞となる場合

図３　ｒと推定精度の関係

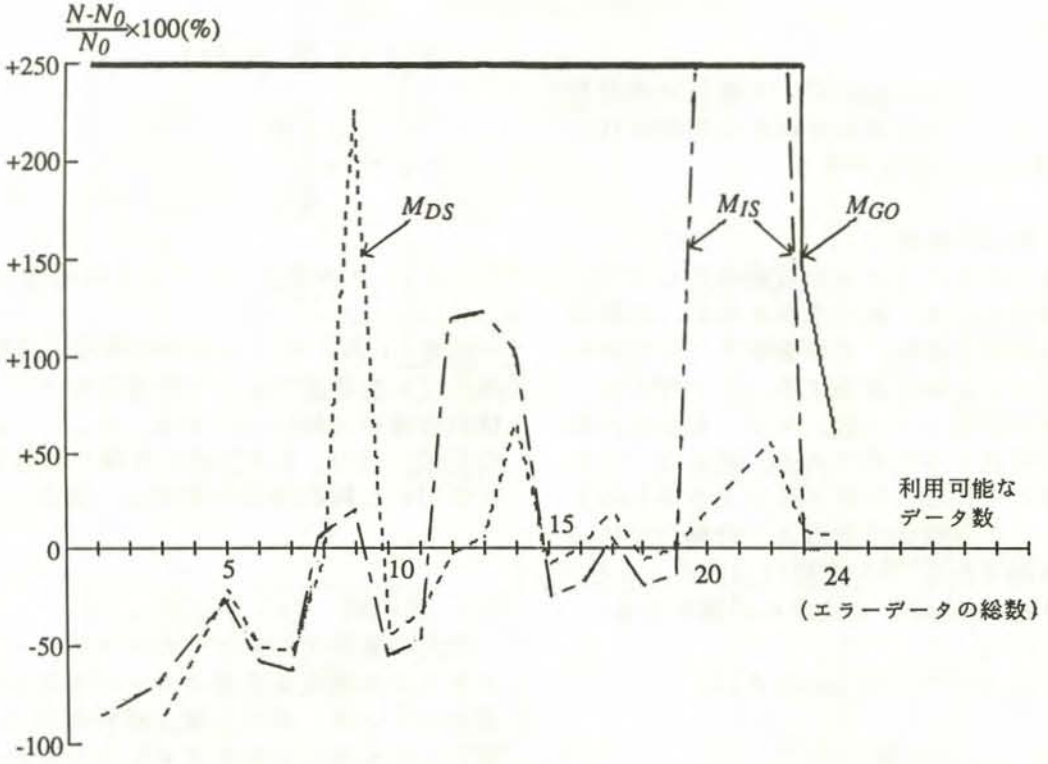


図4　利用可能なデータ数と推定精度の関係

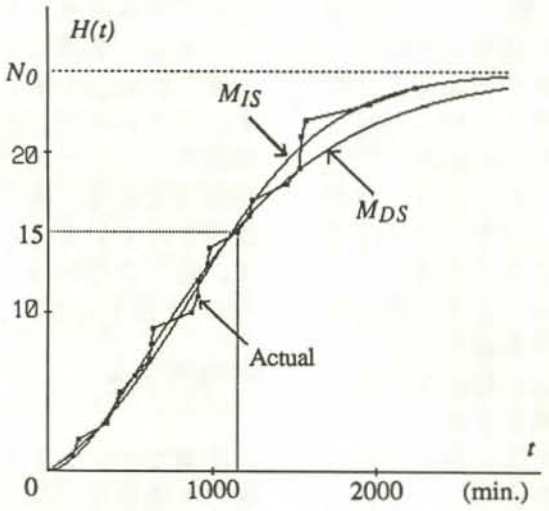


図5　利用可能なデータ数が15個の場合（x132）

6. モデルM_{HE}における原点ずらし

ここでは，モデルM_{HE}に対する新しい適用方法を提案する．この方法は信頼度成長曲線$H(t)$の原点の取り方に特徴がある．

6.1 M_{HE}とM_{GO}の比較

M_{HE}は幾つかのクラスタから構成されるプログラムに適用される．各クラスタ毎に，指数型信頼度成長曲線を求め，それを単純に加え合わせてプログラム全体を評価する．その理由は，クラスタ毎に初期エラー数，エラー発見率が異なることを考慮するためである．例えば，プログラムが新しく作成されたクラスタと再利用されたクラスタから構成されるような場合にM_{HE}モデルは有効である[6]．文献[6,11]によると，M_{HE}における$H(t)$は次のように定義される．

$$H(t)=\sum_{i=1}^{n} N_i\{1-\exp(-\phi_i t)\}$$

n ：クラスタの数
N_i：クラスタiの初期エラー数
ϕ_i：クラスタiのエラー発見率

表6にM_{HE}の適用結果を示す．表6中のN_P，ϕ_Pは構文解析部の初期エラー数，エラー発見率を表す．N_G，ϕ_Gはコード生成部の初期エラー数，エラー発見率を表す．表6と表4を比較すると，このデータに関する限り，M_{GO}がM_{HE}より優れていることが分る．1つの理由として，エラーをクラスタ毎に分けたため，エラー数がパラメータを推定するのに十分なだけの大きさでなくなってしまったことが考えられる．

しかし，筆者らはプログラムテストの開始時刻がクラスタ毎に異なることが考慮されていないことに注目する．これがM_{HE}の精度を下げる主要な原因であったとする立場をとる．

注意1:この実験の場合，クラスタ毎にテスト開始時刻を測ることが可能であった．

6.2 原点ずらし

M_{HE}では，クラスタ毎にN_iとϕ_iを求め，求まった信頼度成長曲線$H_i(t)$を単純に加え合わせる．このとき，各信頼度成長曲線の原点は同一であると仮定している．

これに対し，提案する方法では次に示す様に，クラスタ毎に信頼度成長曲線$H_i(t)$の原点が異なるものと考える．こうして$H_i(t)$を加え合わせると，信頼度成長曲線$H(t)$はS字型モデルの曲線に似てくる．

$$H(t)=\sum_{i=1}^{n} H_i(t)$$

$$H_i(t)=\begin{cases} 0 & \cdots t<T_i \\ N_i\{1-\exp(-\phi_i(t-T_i))\} & \cdots t\geq T_i \end{cases}$$

T_i：クラスタiのテスト開始時刻

注意2：各クラスタ毎の信頼度成長曲線の原点のズレを考慮することの重要性について，同様の指摘が文献[11]にある．但し，上述の手順の記述，及び，6.3で述べる様に具体的なデータを用いた解析例等の報告は，筆者らの知る限り，ない．

6.3 適用例

今回の実験で対象としたプログラムは2つのクラスタ（構文解析部とコード生成部）から構成されている．更に，構文解析部が（エラーを含むことを許したとしても）少なくとも動作するまでコード生成部のテストはできない．従って，構文解析部とコード生成部が動作し始めた時刻をコード生成部のテスト開始時刻とみなす．一方，構文解析部のテスト開始時刻については，プログラムテキストの更新記録と各学生へのインタビューを通じて決定した．

表7にM_{HE}の新しい適用方法による結果を示す．表7中のT_Gは，構文解析部のテスト開始時刻を0とした時の，コード生成部のテスト開始時刻である．表6と比べると，Aとpの値が改善されていることが分かる．学生x132に関して，適用方法の違いによる信頼度成長曲線$H(t)$の差を図6に示す．

7. あとがき

本稿ではソフトウェア信頼度成長モデルの実験的評価を行った．実験データは言語Pascalの部分集合に対してコンパイラを作成する学生実験から収集したものである．一般的なモデル評価実験に比べると，プログラムテストの期間が短く，作成されるプログラムの大きさも小さい．また，エラーの発見時刻も分単位で，詳細なものである．

次に，本稿で得られた成果をソフトウェア開発の現場での応用について述べる．

(1) テスト終了時点でのソフトウェアの評価はソフトウェアの品質保証という観点から重要である[4]．3.の結果より，評価に用いるモデ

表6　モデルM_{HE}の適用結果

		x125	x132	x133	x134	x136
初期エラー数　N_0		32	25	23	28	28
M_{HE}	N_P	117.8	18.9	17.9	7.6	36.6
	$\phi_P(\times10^{-3})$	0.29	0.52	1.50	0.66	0.19
	N_G	3.0	27.8	∞	21.7	∞
	$\phi_G(\times10^{-3})$	19.75	0.26	0.00	0.82	0.00
$\bar{A}$- ∞ p-1/5	A	277.4%	87.0%	∞	4.7%	∞
	κ	×	×	×	○	×

表7　新しい方法によるモデルM_{HE}の適用結果

		x125	x132	x133	x134	x136
初期エラー数　N_0		32	25	23	28	28
M_{HE}	N_P	117.8	18.9	17.9	7.6	36.6
	$\phi_P(\times10^{-3})$	0.29	0.52	1.50	0.66	0.19
	N_G	3.0	12.6	∞	20.4	34.1
	$\phi_G(\times10^{-3})$	19.75	1.31	0.00	1.39	0.35
	T_G (min.)	0	400	100	300	150
$\bar{A}$- ∞ p-2/5	A	277.4%	26.3%	∞	0.1%	152.5%
	κ	×	○	×	○	×

ルとしてはS字型モデルのM_{DS}とM_{IS}が有望である．M_{IS}はパラメータrの値の決定が一般に困難なため，M_{DS}の採用がより妥当である．

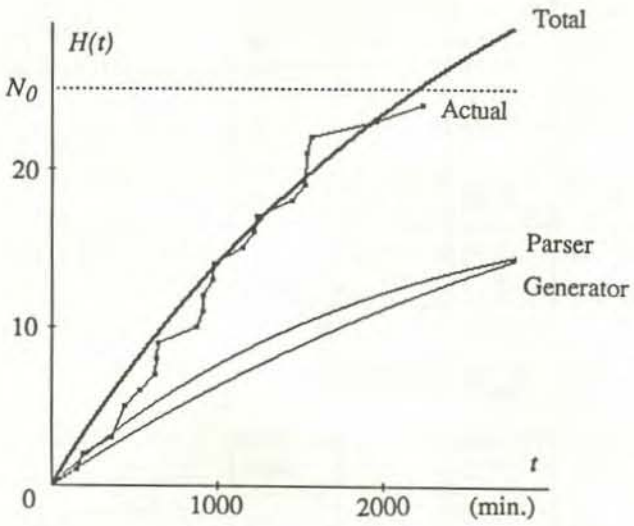
(2) 4.における実験結果より，M_{IS}のパラメータrの値を求める手順は次のようになる．先ず，M_{DS}を用いてN_0の推定値を求める．4.2より，rと推定精度$(N-N_0)/N_0$の間の関係は直線で近似できるので，最適なrの値は横軸を横切る点として求まる．

(3) 5.で議論したテストの進捗に伴うソフトウェアの評価はテスト工程の管理に応用できる．評価結果（Nの値）を過去の同種のプロジェクトの場合と比較して，テスト工程を動的に改善していくことができる．但し，残存エラー数は，テストがある程度進まないと推定できないし，突発的に推定値が∞となることがある．従って，ある時点での評価結果だけではなく，テスト工程全体の流れを見て判断する必要がある．

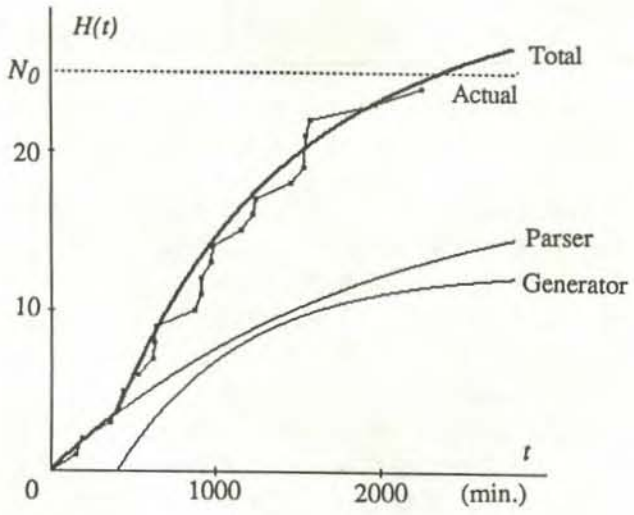
(4) 設計手法に基づいてプログラム開発を行うと，プログラムは一般に幾つかのクラスタから構成される．もし，各クラスタのテスト開始時刻の差が明確であれば，6.で提案した方法の様に，評価モデルの適用に当たってその点を考慮する必要がある．今後の課題としては，各テスト作業がどのクラスタを対象としているかも考慮する必要があると考えられる．

今後，新年度の学生実験を対象として，ソフトウェア信頼度成長モデルの実験的評価を行う予定である．実験では，プログラムの設計，コーディング段階のエラーデータに対するモデルの適用，HEモデルに対して提案した方法の他のモデルへの応用などについても検討する．

謝辞：本報告の作成において，有益な御助言を頂いた日本アイ・ビー・エム㈱東京基礎研究所大場充氏に深謝致します．更に，本評価において快く被験者を引き受けて頂いた大阪大学基礎工学部情報工学科の方々に感謝致します．



(a)従来の方法による適用結果



(b)新しい方法による適用結果

図6　モデルM_{HE}の信頼度成長曲線（x132）

参考文献

[1] A.L.Goel, and K.Okumoto, "Time-dependent error detection rate model for software reliability and other performance measures," IEEE Trans. Rel., vol. R-28, pp.206-211, Aug. 1979.

[2] 梶山 他,"テストの習熟を考慮したソフトウェア信頼度成長モデル," 情報処理学会第25回全国大会, 1E-3, pp.401-402, 1982.

[3] K.Kishida et al.,"Quality - assurance technology in Japan," IEEE Software, vol.4, no.5, pp.11-18, Sep. 1987.

[4] 小室,中山,"テスト支援ツールの実践的応用（各種エラー発生成長モデルの適用)," ソフトウェア・シンポジウム'87 論文集, pp.285-295, 1987.

[5] K.Matsumoto et al., "Experimental evaluation of software reliability growth models," FTCS-18, to appear.

[6] M.Ohba,"Software reliability analysis models," IBM J. Res. Develop.,vol.28, no.4, pp.428-443, July 1984.

[7] 大場 他,"ソフトウェア・エラー推定法の比較,"情報処理学会第25回全国大会, 1E-4, pp.403-404, 1982.

[8] 大場 他,"習熟型ソフトウェア信頼度成長モデル," 情処研報, SW-28-6, 1983.

[9] S.Yamada, M.Ohba and S.Osaki, "S-shaped reliability growth modeling for software error detection," IEEE Trans. Rel., vol. R-32, pp.475-478, Dec. 1983.

[10] S.Yamada, and S.Osaki, "Software reliability growth modeling : Models and applications," IEEE Trans. Software Eng., vol. SE-11, pp.1431-1437, Dec. 1985.

[11] 米原 他,"モジュール構造ソフトウェアの信頼度成長曲線," 情報処理学会第25回全国大会, 1E-2, pp.399-400, 1982.