5
Programs, Cities, Students—Limits to Growth?

M. M. Lehman
University of London

The battlefield of programming

In an invited lecture to the 1971 IFIP Congress [Randell 71b*], Brian Randell of the University of Newcastle typified the situation current in the design, implementation and maintenance of computer software systems by showing a slide (Plate I)[1] of a medieval battlefield with ranks aligned but carnage abounding and devastation everywhere.

The situation today is really very little different. The programming systems world is a little older, perhaps a little wiser, but has in its practice not learnt much from the experiences of the last ten years. Many still ride bravely into battle (Plate IIa) determined to master the system that they wish to create or maintain. Most in one way or another fall by the wayside (Plate IIb). And programming expenditure goes up and up and up.

The total picture is however not completely bleak. Some dragons have been slain (Plate III). The need for a discipline of software engineering is recognized [Naur 69b*]. The formulation of concepts of programming methodology exemplified by Dijkstra's concept of structured programming [Dijkstra 72d*] strikes at the roots of the problem. The realization that a program, much as a mathematical theorem, should and can be provable and proven correct as it is developed and maintained [Dijkstra 68e*] and before its results are used will ultimately change the nature of the programming task and the face of the programming world.

[1] My thanks are due to the management of the Alte Pinakothek Museum in Munich for permission and facilities to produce the first three slides, to Mr. P. Young of the Victoria and Albert Museum for the loan of slide four and to the Directors of both museums for permission to reproduce the slides in the published version of the lecture. Also to Professor G. Seegmueller for his assistance in the preparation of the slides.

D. Gries (ed.), *Programming Methodology*

Randell's depiction of the software crisis. (Detail from *Die Alexanderschlacht*, Albrecht Altdorfer, by courtesy of the Alte Pinakothek Museum, Munich.)

Plate II

(a) Ride into Battle. (Detail from *Die Alexanderschlacht*, Albrecht Altdorfer, by courtesy of the Alte Pinakothek Museum, Munich.)

(b) Fallen by the Wayside. (Detail from *Belagarung Von Alexia*, Feselen, by courtesy of the Alte Pinakothek Museum, Munich.)

Some Dragons Slain. (Panel from *Altar-piece with scenes from the life of St. George* attributed to Marzal de Sas, by courtesy of the Victoria and Albert Museum, London.)

Plate IV

(a) The Top-down Approach. (Photo-montage on *Die Alexanderschlacht*, Albrecht Altdorfer, by courtesy of the Alte Pinakothek Museum, Munich.)

(b) The anti-regressive ant.

Clearly these developments are of fundamental importance. They offer the only long-term solution to the creation of the masses of programming material that the world appears to require; or at least that computer manufacturers and software houses think the world requires. Nevertheless we must face the fact that progress in mastering the science of program creation, maintenance and expansion will be painful and slow.

The systems approach

Such progress as is currently being made stems primarily from the personal involvement of the researchers and the developers in the programming process at a detailed level. Often they only tackle one problem area: algorithm development, language, structure, correctness proving, code generation, documentation, testing. Others view the process as a whole but are still primarily concerned with the individual steps that, together, take one from concept to computation. And this type of study is clearly essential if real insight is to be gained and progress made.

But application of the scientific method has achieved progress in revealing the nature of the physical world by also pursuing a course other than studying individual phenomena in exquisite detail. A system, a process, a phenomenon, may be viewed in the first place from the outside, observing, clarifying, measuring and modelling identifiable attributes, patterns and trends. From such activities one obtains increasing knowledge and understanding based on the behavior of both the system and its sub-systems, the process and its sub-processes. Following through developing insight in structured fashion, this outside-in approach in due course leads to an understanding of, and an ability to control, the individual phenomena but in the context of their total environments.

In terms of the previous analogy one may overfly (Plate IV) a battle, study it using all available observational tools. Thereby one would observe its ebb and flow identifying the location, global characteristics and, on closer and closer inspection, the nature of the main points of advance, or of chaos and destruction. Having succeeded in this, one may hope to better understand, and hence modify and control, what is going on.

In my present area of interest and concern I have adopted such a structured analysis approach to study the programming process. I shall be showing you samples of data that support this systems approach to the study of the process. One need not search too hard for its conceptual justification. A programming project can involve many hundreds of people and many tens of managers. Many millions of pounds or dollars may be spent on it. Thus individual decisions of almost any form make very little impact on the overall trend. It is really the inertia of people and habits, the momentum of practices and budgets, the general smoothing effect of organizations, that determine the rate of progress and fate of a project.

The roots of the study

Some years ago I undertook a study of the programming process in one particular environment [Lehman 70*]. As part of the study I obtained project statistics of a programming system which had already survived a number of versions or releases. The data for each release of the system included measures of the size of the system, the number of modules added, deleted or changed, the release dates, information on manpower and machine time used and costs involved in each release. In general there were large, apparently stochastic, variations in the individual data items from release to release. But overall the data indicated a general upward trend in the size, complexity and cost of the system and the maintenance process (Figure 1).

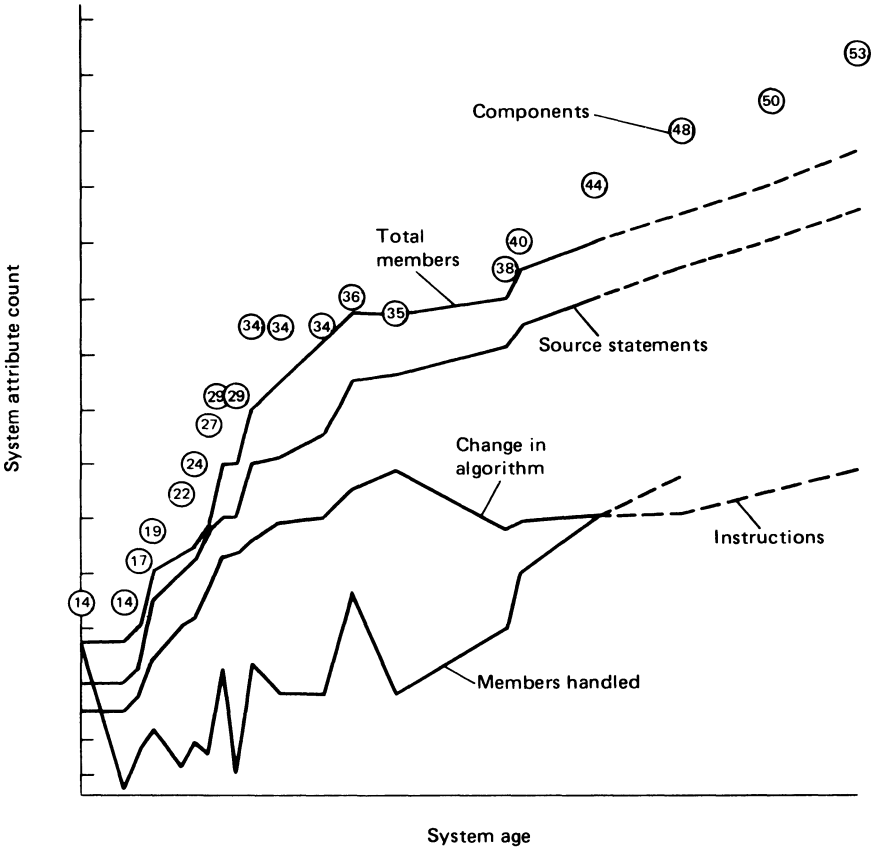


Figure 1 Early data.

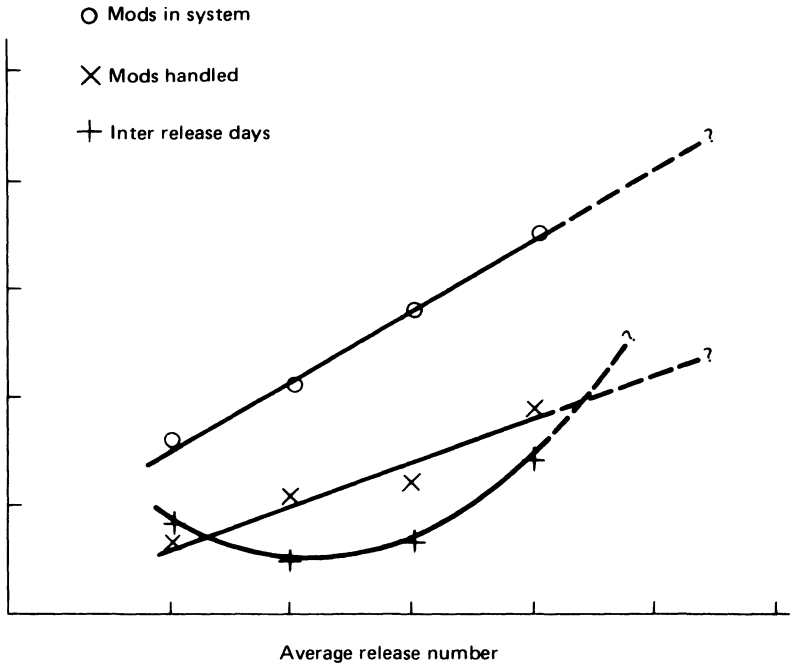


Figure 2 Initial data averaged.

As a first step in my study of this data I averaged the various parameters. The intent was of course to expose any specific trends. And expose them it did. When the averaged data was plotted the previously erratic data had become strikingly smooth (Figure 2).

Some time later additional data (Figure 3) confirmed previous suspicions of nonlinear, possibly exponential, complexity growth. Moreover, extrapolation of the plots suggested further growth trends significantly at odds with the then current project plans. The data was also highly erratic (Figure 4) with major but apparently serially correlated (Figure 5) fluctuations from release to release. Nevertheless almost any form of averaging led to the display of very clear trends (Figure 6). Thus it was natural to apply uni- and multi-variate regression and auto-correlation techniques to fit appropriate regression and time-series models to represent the process for purposes of planning, forecasting and improving it in part or as a whole. In general there was definite evidence that one might consider a software maintenance and enhancement project as a self-regulating organism subject to apparently random shocks, but overall obeying its own specific conservation laws and internal dynamics.

All in all these first observations encouraged me to search for laws that governed the behavior, the dynamics, of the meta-system of organization, people and program material involved in the creation and maintenance process, in the evolution of programming systems.

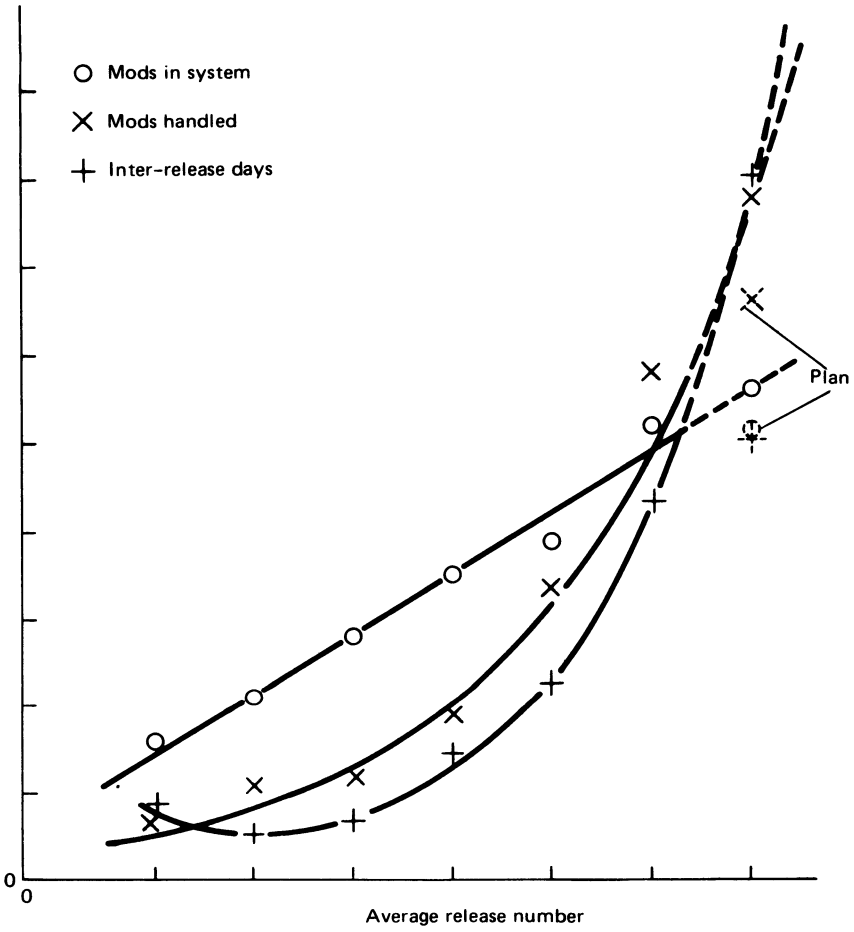


Figure 3 Later data.

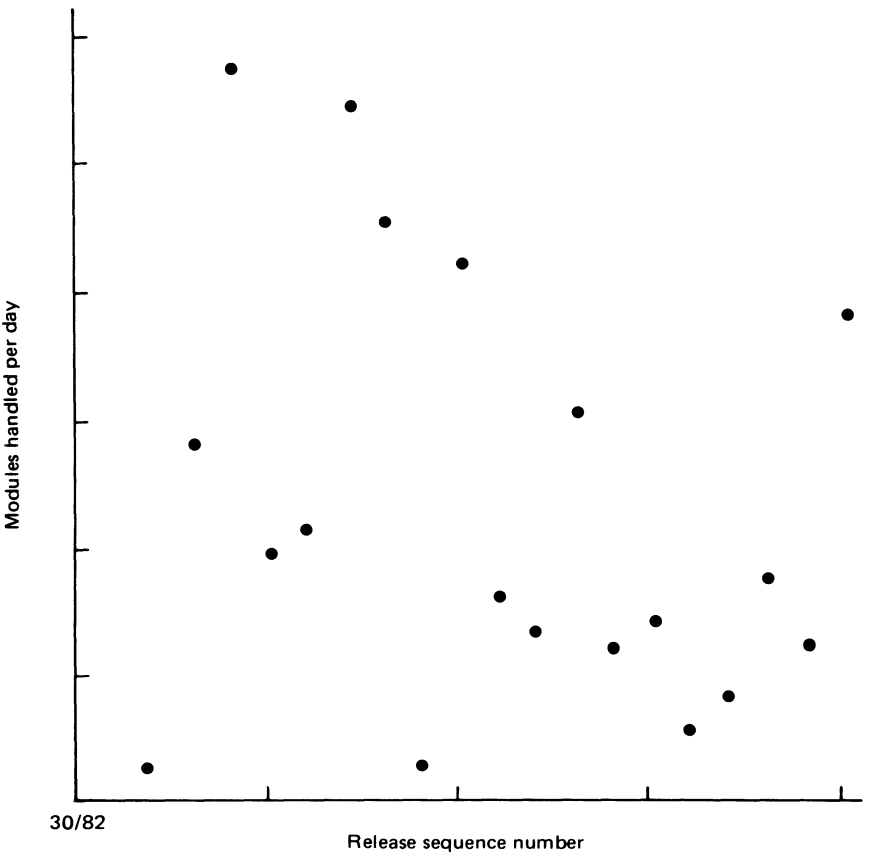


Figure 4 Scattered data.

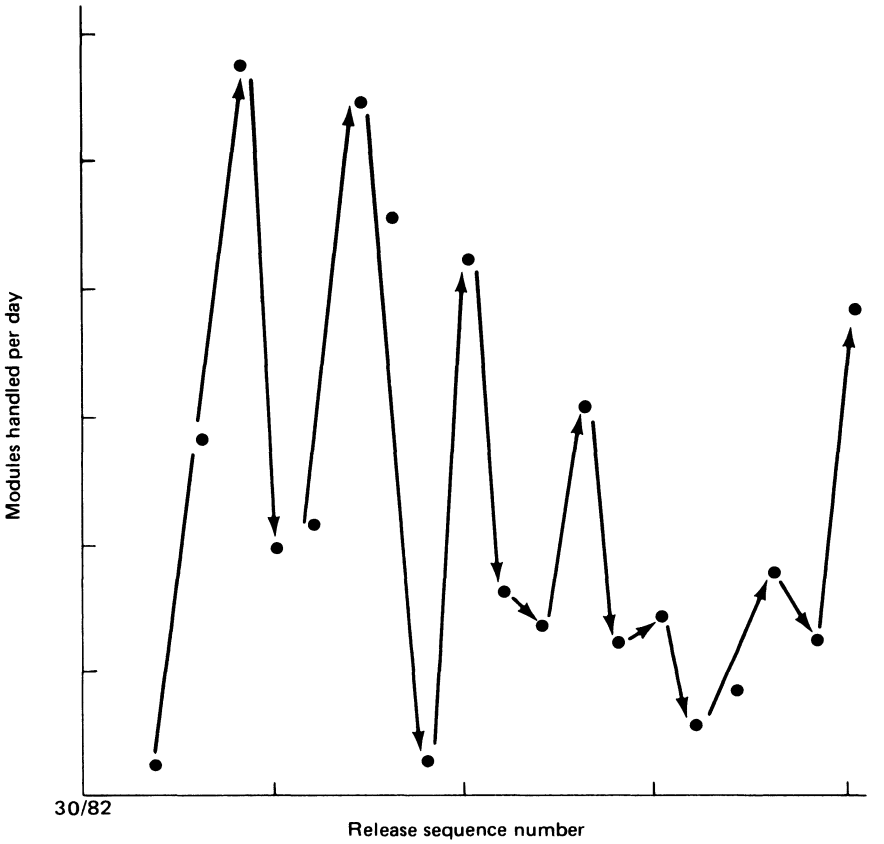


Figure 5 Serial correlation.

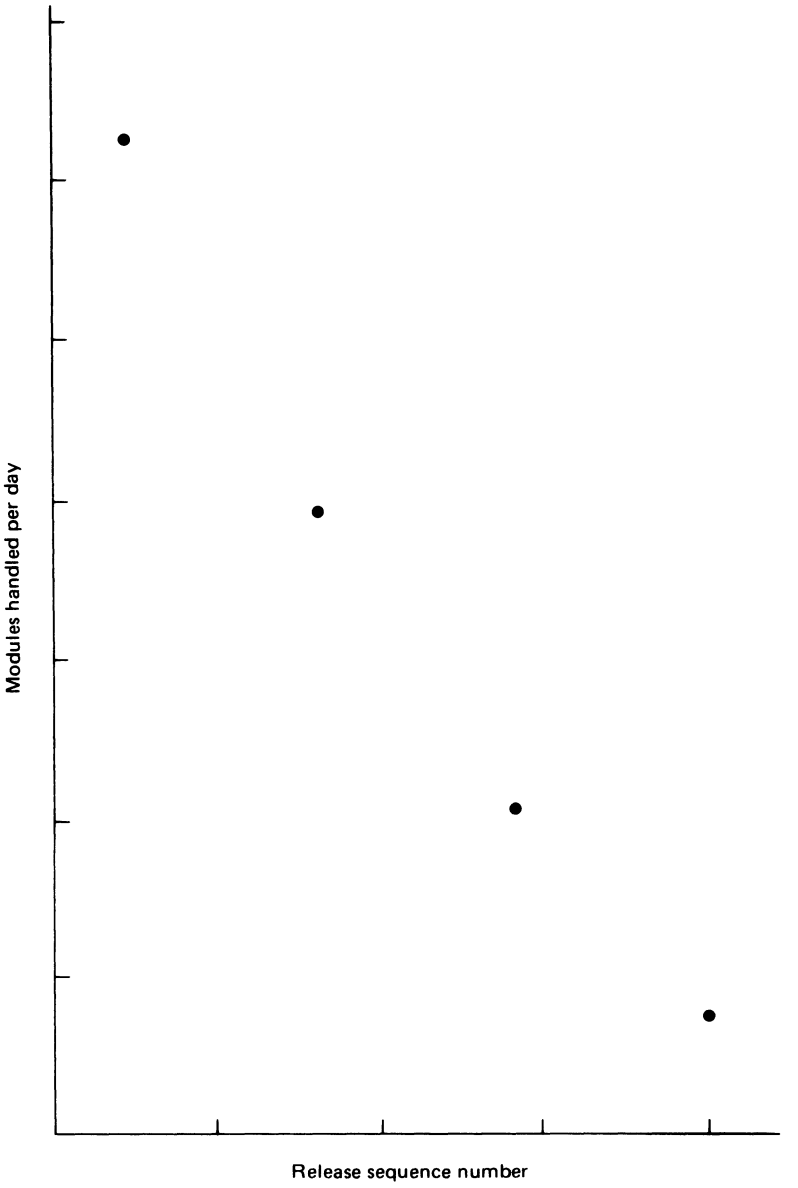


Figure 6 Overall smoothness.

Laws of programming-system life dynamics

It is perhaps necessary to explain here why I repeatedly refer to the continuous creation, maintenance and enhancement of programming systems. It is the actual experience of all who have been involved in the utilization of computing equipment and the running of large multiple-function programs that the latter demand continuous repair and improvement. Thus I postulate as a *First Law* (Figure 7) of Program Evolution Dynamics:

THE LAW OF CONTINUING CHANGE

Software does not face the physical decay problems that hardware faces. But the power and logical flexibility of computing systems, the extending technology of computer application, ever-evolving hardware and the pressures for the exploitation of new application areas all make users demand, and manufacturers encourage, continuous adaptation of programs to keep in step with increasing skill, insight, ambition and opportunity. Thus a programming system undergoes never-ending maintenance and development, driven by the potential difference between current capability and the demands of the environment.

As a system is changed it inevitably becomes more complex and unmanageable. Hence I also postulate a *Second Law*:

THE LAW OF INCREASING ENTROPY

This law too is supported by universal experience and in conjunction with the evidence deduced from such programming project data as has been available and studied, has led me to the formulation of a *Third Law*:

THE LAW OF STATISTICALLY SMOOTH GROWTH

The system and its metasystem, the project organization developing it, constitute an organism constrained by conservation laws. These may be locally overcome, but they direct, constrain and control the long-term growth and development patterns and rates. It is the study of one aspect of this third law, and its generalization, that forms the underlying theme of the remainder of my talk.

I Law of continuing change

A system that is used undergoes continuing change until it becomes more economical to replace it by a new or restructured system

II Law of increasing entropy

The entropy of a system increases with time unless specific work is executed to maintain or reduce it

III Law of statistically smooth growth

Growth trend measures of global system attributes may appear stochastic locally in time and space but are self-regulating and statistically smooth

Figure 7 Laws of program evolution dynamics.

Statistical models of system growth

These laws, as formulated now, have gradually evolved as we have pursued our study of the programming task. In the first instance my observations simply led to the conception of an area of study which, at the time, we termed Programming Systems Growth Dynamics [Lehman 71*] but which we now prefer to refer to as 'Evolution Dynamics'. In close association with a colleague, L. Belady (who I am happy to say is here this evening as an SRC-sponsored Senior Visiting Research Fellow) it has also led to the development of statistical and theoretical models, such as those that I shall discuss, and to an ever-increasing understanding of the nature and dynamics of the programming process.

Let me give just one, very significant, illustration of the statistical models. Figure 8 plots the size of the system as measured in modules. Clearly the growth measured as a function of release numbers has been strictly linear. This growth is really very steady. However closer examination of the plot reveals a superimposed ripple (Figure 9). To control engineers this may suggest a self-stabilizing, multi-loop feedback system. Positive feedback arises, for example, from the desire of users to get more

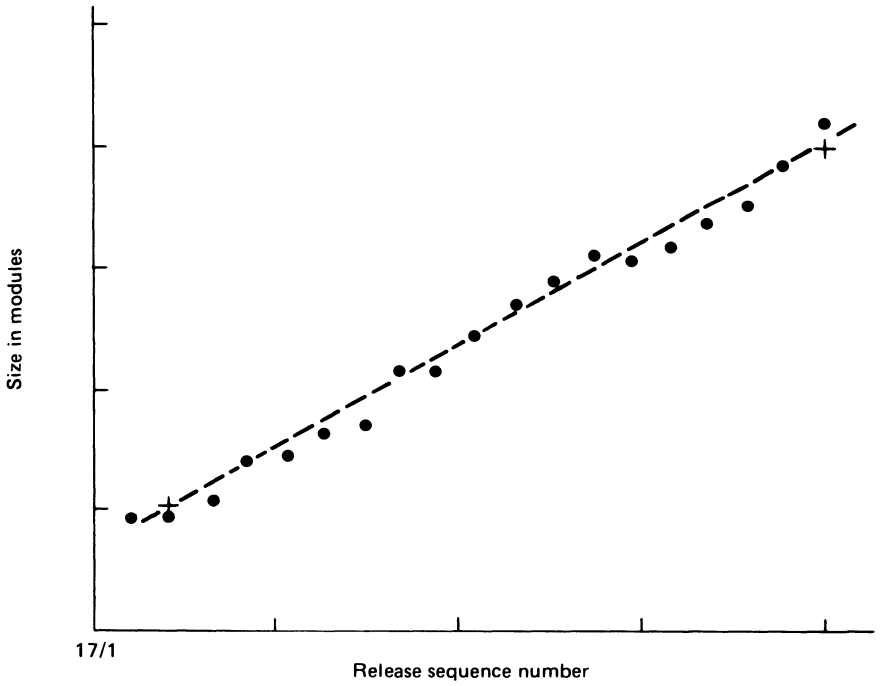


Figure 8 Linear growth.

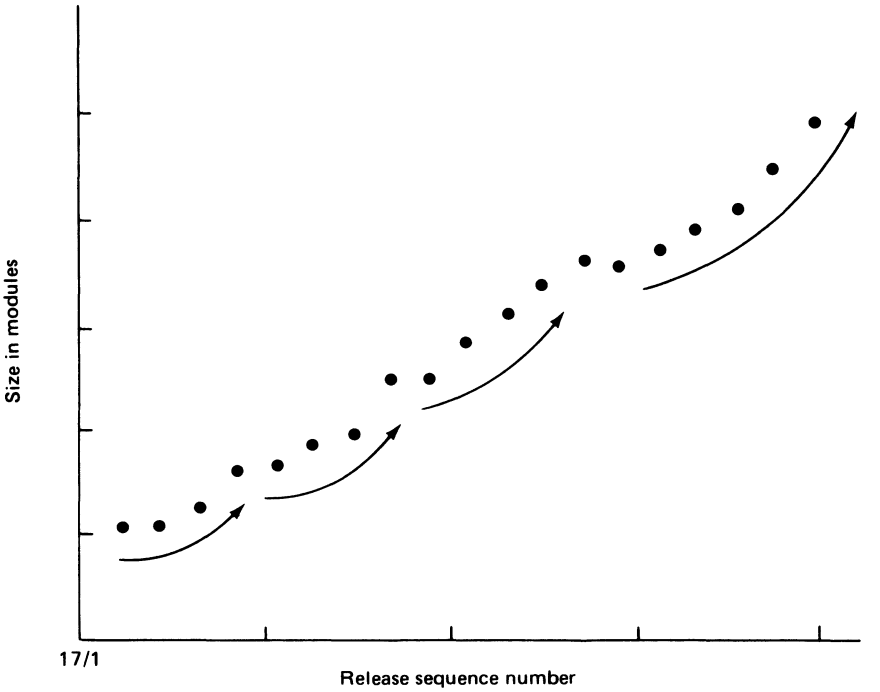


Figure 9 Growth cycles.

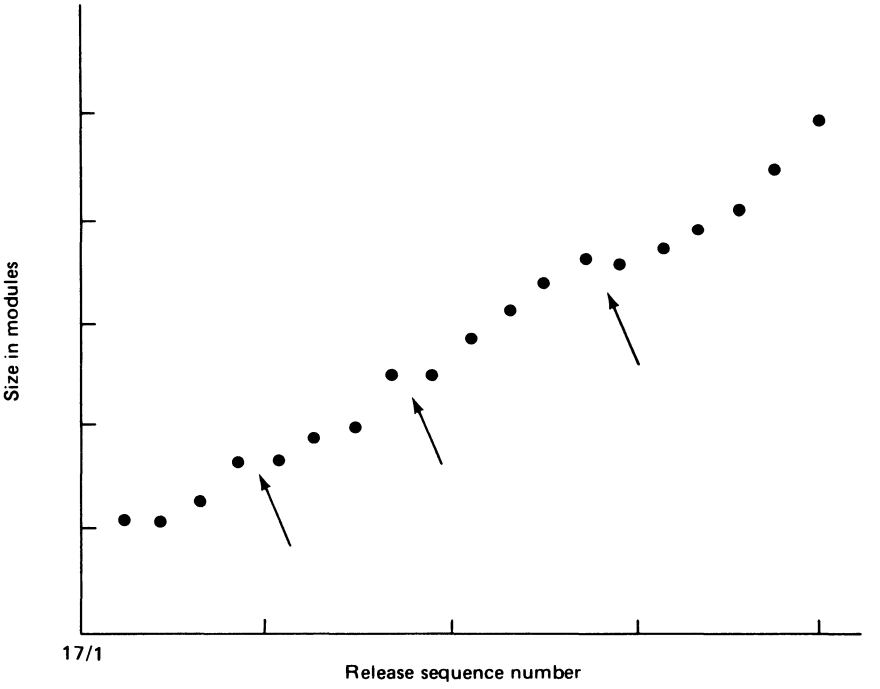


Figure 10 Clean-up points.

function, more quickly. This creates the pressures that cause the rate of systems growth to increase.

But a compensating negative feedback every now and again causes a decline (Figure 10). As attempts are made to speed up the growth process, design and implementation may become sloppy, short cuts are adopted, testing is abbreviated, documentation falls behind, errors are made and the fault rate builds up. Changes to the code become progressively more difficult. Project and system entropies have become large, their structures

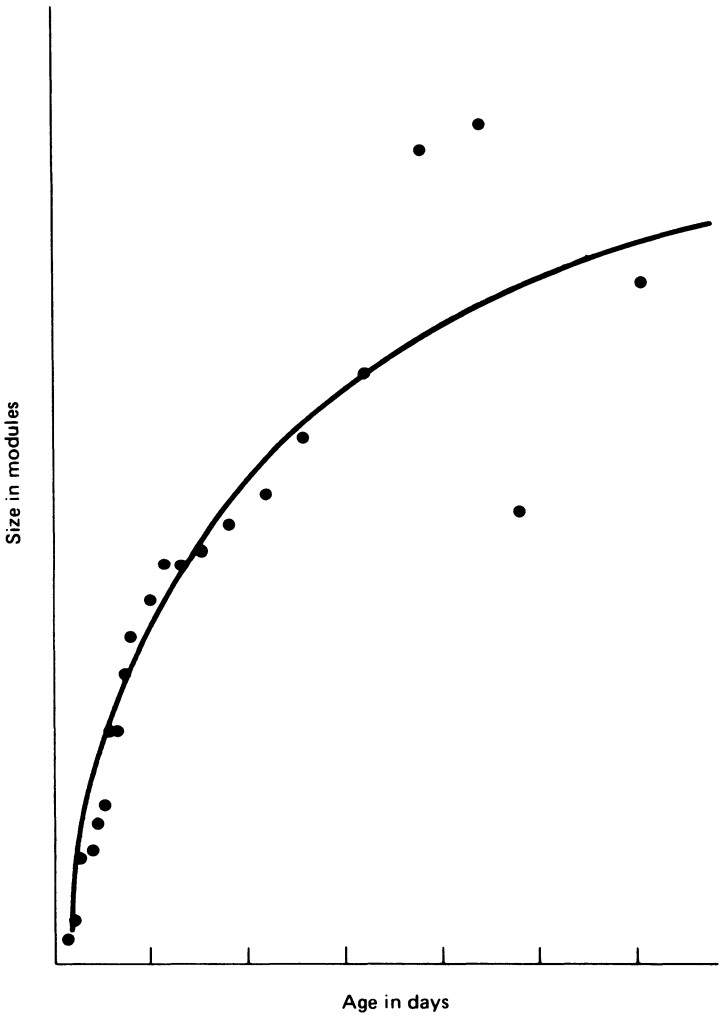


Figure 11 Limit to growth?

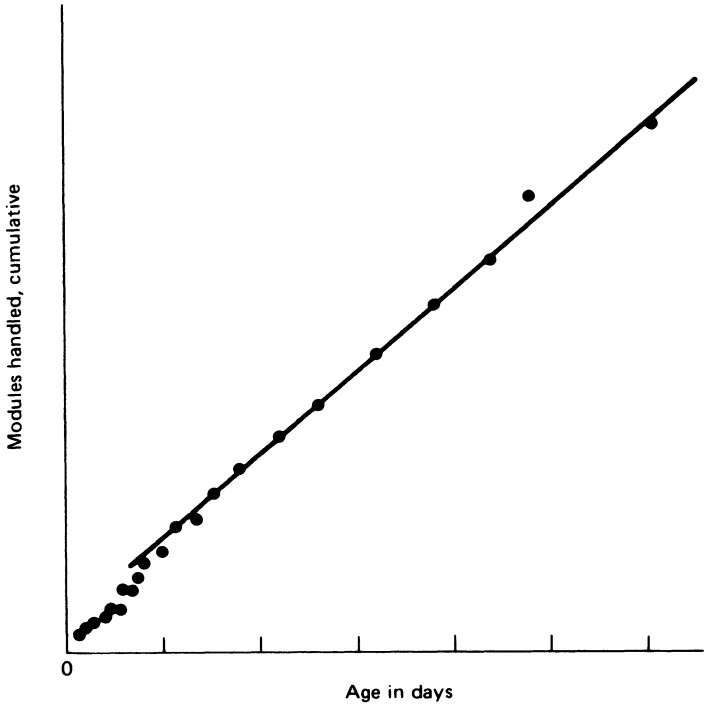


Figure 12 Constant rate of work-output.

have become hazy. The net result is that the growth rate declines. Both systems must be cleaned up, their structures improved, their entropy reduced, before enhancement of the system can be resumed.

Incidentally it may appear strange that apparently valid statistical models are obtained using an arbitrarily assigned 'Release Number' as one of the variables. Its use as a 'pseudo-time' variable can be justified by the 'Principle of Parsimony' [Box 70], the simplicity and regularity of the models that its use yields. A more fundamental justification follows from a fuller understanding of the role of the system release point as a stabilizing or anchor point in the programming process and for the programming organization. In fact what we have termed the 'release number' must really be viewed as the release '*sequence*' number, which may well differ from the former and which, quite naturally, forms a basis for the time-series type of analysis [Cox 66]. And there can be no question that results to date fully substantiate the claim that release-based project data provides a useful and powerful planning tool. Used in conjunction with real-time based models (Figure 11, 12) it is also beginning to yield insight into the programming process, insight that will eventually permit its improvement.

The macro models

The Yorktown model

The statistical analysis represents, at most, a tool to guide our understanding, and hence mastery, of the intellectual and organizational effort that really underlies the programming process. In parallel with that analysis Belady and I looked at fundamental aspects of the programming process, searching for representative theoretical models. It is not possible to elaborate on the details of the resultant sequence of models in the present lecture. I will nevertheless show you the forms of two of the earlier ones, without however interpreting all their terms, to give some idea of the approach taken.

As a first approach [Lehman 71*] we developed a model to represent the communication cost of such a project. We showed [Lehman 72*] that in general the cost W_{i+1} of taking a programming system from release i to release $(i+1)$ can be represented as:

$$W_{i+1} = W(B_i) + wH_i^2 \cdot 2^{2G_i \cdot i - DAL_i} \qquad [1]$$

Briefly the coefficient H represents the entropy, the degree of unstructuredness, of the program itself. G represents the entropy of the project. It measures the communication effort required at all times for coordination between the groups and individuals maintaining and updating the system. Finally the factor DAL represents the state of knowledge of systems and meta-system capability structure and content, the correctness, completeness and accessability of system and project documentation.

This macro model is static and does not express any controlling feedback relationships. It merely indicates that one source of exponential cost growth is the ever-increasing difficulty of ensuring that changes to the code are compatible with its past definition and behavior. Changes to the system today must not undo yesterday's work or prevent a repeat of yesterday's usage. Nor must they conflict with activities occurring elsewhere in the organization, today.

Any tendency to diverge may be minimized and controlled by creating and maintaining a well-structured system (H small) and/or by having a well-structured, strongly communicating, project organization (G small). Additionally, specification and documentation accuracy and program clarity must be maintained, so as to ensure that all questions about the meaning, intention or effect of code can easily and unambiguously be answered and, ideally, so that the code can be proven correct. The exponential growth trend would be suppressed given perfect system structure, perfect project structure or perfect documentation but in practice these cannot be achieved or maintained.

The Berkeley model

We now leave this primitive model and turn to one that displays the dynamics of the process. A dynamic macro model may be developed directly [Lehman 72b*], applying the techniques of structured analysis, to yield as a simple example at the highest level and where all parameters are assumed to be functions of time:

$$W = u \cdot F + M \cdot v \cdot K \cdot F + N \cdot w \cdot E + P \cdot x \cdot R \qquad [2]$$

In any programming project there will be activity related to the design and creation of new code and the modification of existing code. This is represented in [2] by a factor F. All such activity must be accompanied by additional activity, $K \cdot F$, to document and record it, and both activity measures must be multiplied by expenditure rates of factors u, v say, to yield the actual cost rate or cost terms.

There is also the activity represented by the exponential term in the previous model and now abbreviated to E, yielding a model term $w \cdot E$. Finally, in any project, there should be concern to improve the quality of the product and the productivity of its participants. The expenditure rate or expenditure of such activity is modelled by a term $x \cdot R$. This therefore represents that part of the budget applied to methodology improvement and tool development.

We observe that the level of productive activity F and its division between, for example, repair (F_1) functional improvement (F_2) and the creation of new capabilities (F_3) is a management judgment based on fault rates, business considerations and pressures from users. But once their average rate or level has been set, that of the activities represented by the other terms is predetermined if the system is to remain healthy—that is, usable, maintainable and enhanceable.

In practice, however, under the pressures of continuous demand and imminent completion dates, the work whose neglect has no immediate consequence is pushed aside. Thus the average level of activities represented by the last three terms will often fall below the level required to match program creation and modification. This reduction, a consequence of conscious or unconscious management decisions, is reflected in the model by the addition of management factors M, N, P that are, in general, less than 1.

The inter-parameter relationships

We have noted that all the variables and parameters in [2] are themselves time dependent. Additionally there will be time-dependent relationships between them. A complete model must express these relationships. Thus, as an example, if documentation is neglected ($M < 1$), the cost term u will

increase in the future as knowledge of the state of the system declines and it becomes increasingly difficult to understand the content, intention or meaning of a piece of code. Moreover errors or defects in the system will increase, that is the ratio of repair activity (F_1) to enhancement (F_2 and F_3) will increase, an effect which is critical (and observable) in the dynamics of the process. Possible sets of relationships, the families of differential equations they lead to and the relationships between the solutions of these equations, available project and program data and our understanding of the programming process, are now being studied in our SRC supported research project under the title Systems Engineering (Growth Dynamics), and in a project 'Program evolution dynamics' led by L. Belady at the IBM Yorktown Heights Research Laboratories.

Progressive activities

Much more could be said about this family of models. For now, however, I draw attention to just one of its characteristics. If we examine the four terms of [2] we find that the first is concerned with activity undertaken because of its assessable value to the organization and the user. It produces code, hopefully usable code. We have added as it were to the store of potential energy in the system, to its power. We have increased the value of the system. I term activity of this type *progressive*.

Anti-regressive activities

The other three terms on the other hand do not, by and large, have a direct or immediate effect on the power or value of the system. System documentation, for example, must be undertaken while the work is being done or shortly thereafter. But it will be used only at some future time when it becomes necessary to modify the system. If it is there, well and good. If not trouble lies ahead. The system will be difficult to repair, to change. It may not be possible to keep the system in harmony with a changing environment. Relative to its environment, it will have begun to decay.

The prime purpose of the expenditure represented by the E term is to prevent the insertion or perpetuation of a fault in the system that will, at some future time, result in undesirable or incorrect operation. Ideally it relates to the activity that would be required to re-prove correctness of the system each time it is changed. Since in the present state of the art this cannot, in general, be done, the term can equally be interpreted to represent testing activity which serves as a poor, but as yet essential, substitute. Thus all in all the term models the effort to minimize the number of undiscovered faults in the system. It too represents an investment in the future.

Finally the fourth term, modelling the cost of methodology and tool development, also represents a long-range investment. It is concerned with maintaining a capability to cope with system development and maintenance despite increasing size and complexity.

In general these three elements of the model represent the cost of effort that I term *anti-regressive*. They are concerned not with increasing the value of the system immediately but with the investment of activity today to prevent system unmaintainability, and hence decay, in the future.

Unbalanced productivity growth

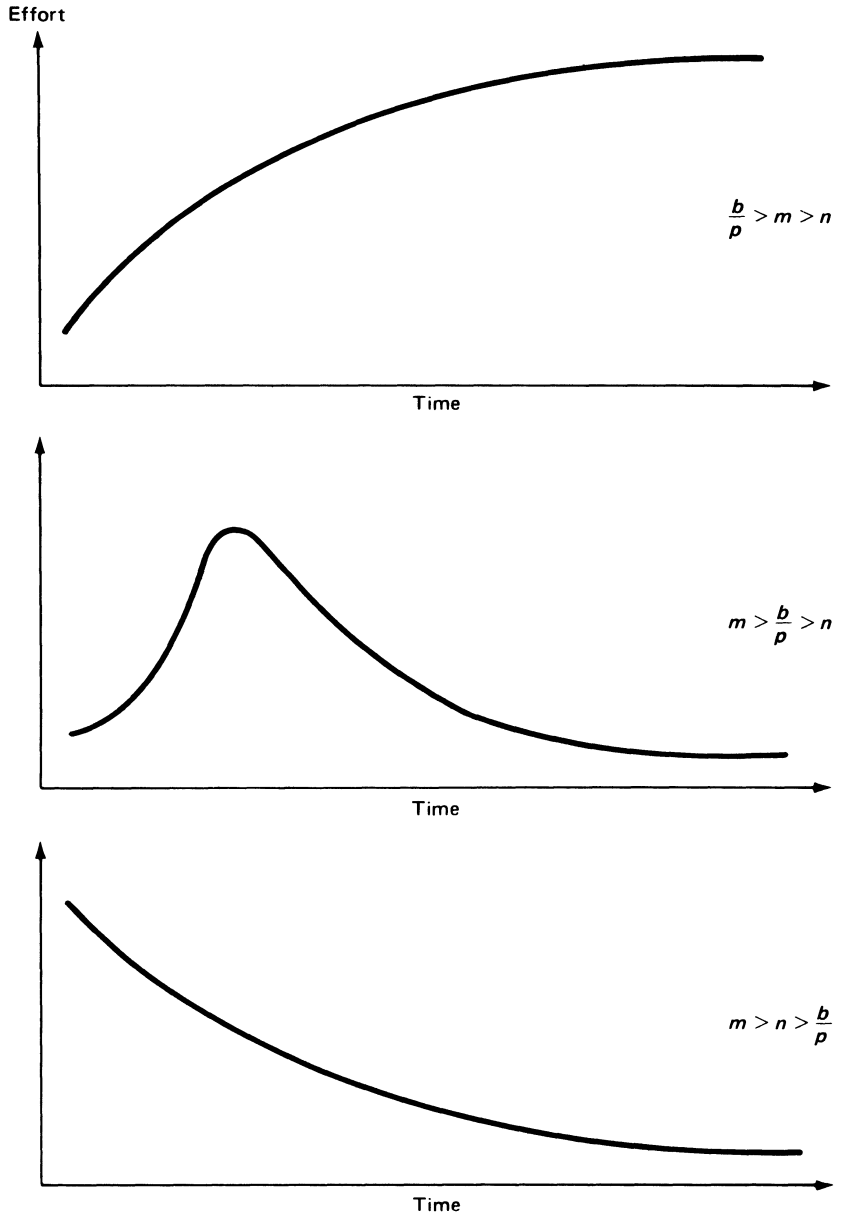
Baumol's Principle and its generalization

It is my thesis that the complementarity of progressive and anti-regressive activity is fundamental to all human activity.

In 1967, [Baumol 67] discussed the consequences of unbalanced productivity growth. His principle conclusion was that areas in which there is an intrinsic barrier to productivity increases must ultimately be priced out of commercial existence. But is this effect limited to instances where the obstacle to productivity increases is intrinsic? Any politician, administrator, manager, or even individual will be prepared to make an investment or an effort now, if the return is immediate or at least demonstrable. However to invest now so that in the future decay can be avoided or some other activity will be more efficient is quite a different matter. Thus there is inherent psychological pressure for productivity to increase faster in progressive than in the anti-regressive areas. I suggest now that a Baumol-like effect will occur also in this case despite the fact that there is no intrinsic barrier to productivity growth.

What is the consequence of this? In general all progressive activity must be accompanied by some anti-regressive activity. As progressive activity and productivity grow so does the demand for anti-regressive activity. But so does the cost of labour in both areas. Therefore more and more anti-regressive activity, costing more and more, is required and until something is done about productivity in the anti-regressive area one of two things may happen. Resources may be diverted from the progressive to the anti-regressive area. Progressive activity must then fall, growth rates decline and ultimately actual recession may set in.

Alternatively anti-regressive activity may be neglected. In that event, inevitably, sooner or later further growth grinds to a halt. In a programming system, for example, this occurs when the fault rate becomes so high that the system must be cleaned up before further developments can proceed, an effect clearly visible in Figures 10 and 11.



$$\frac{w}{v} = \frac{b_0q_0 + (bq_0 + pb_0q_0)t + bpqt^2}{q_0 + (l_0 + l_1)(1 + q_0) + pq(1 + l_1) + p(l_0 + 2l_1)t + p^2t^2}$$

$$m = \frac{l_1b_0}{q_0(1 + l_1) + l_0 + l_1} \qquad n = \frac{b_0(l_1 - q_0l_0)}{q_0 + (l_0l_1)(1 + q_0)}$$

Figure 13

The London model

The preceding discussion suggests a further abstraction of our dynamic model. We may in fact model the dynamics of the programming process as a function of the interaction between progressive and anti-regressive activity, work output and system growth or evolution. By presenting a higher level view of the process, such a model may provide a more concise summary of its time-behavior.

Also by abstracting in appropriate terms, the model may yield a wider interpretation, and therefore be used to describe a more universal phenomenon.

It is unfortunately not possible to present here a precise definition of the concepts of Progressive (P) and Anti-regressive (AR) activities, as would be required to develop such a model. Instead I shall simply outline, in terms of more primitive concepts, a model that results from some simple assumptions about the relationships between them and suggest one consequence that appears to relate to Growth and the Limits to Growth.

The primary assumption will be that productivity increases more rapidly in the progressive areas than it does in the anti-regressive.

With an additional assumption of linear productivity growth, the resultant models of manpower allocation and work output take the form of rational polynomial functions of the time variable t (Figure 13). We cannot

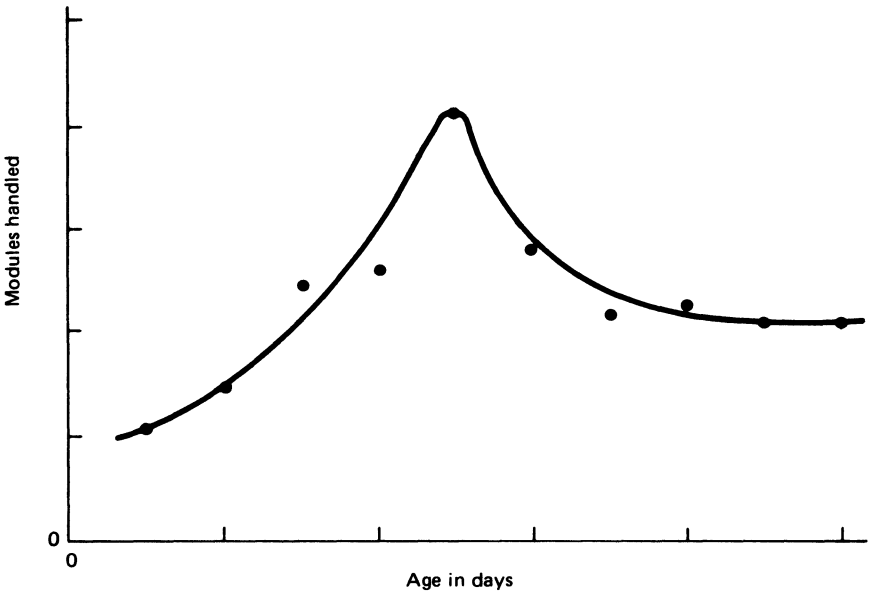


Figure 14 Average work-output rates.

examine these models fully here, but a sample system output shows clearly that, at best, under a set of rather unlikely conditions, the output grows to an asymptotic limit. In practice however output can ultimately be expected to decline. This is entirely due to the progressive and anti-regressive productivity imbalance and can therefore only be mastered by careful control of that balance. And this effect is very real, as can be seen from the behavior of a derived rate-of-work measure taken from actual programming project data (Figure 14).

Generalization

Progressive and anti-regressive activities in city life

The high-level model that has been outlined was suggested by the earlier model of the programming process. I now suggest that it may also be interpreted within the context of sociological and economic systems. In a city, for example, we may identify as progressive (as previously defined) work output that contributes positively to the growth of the standard of living and the quality of life of the community.

But a community also undertakes anti-regressive activities. Collecting garbage, for example, does not increase the value of life in the city, it merely maintains the status quo. It is the neglect of that activity that leads to decay. Similarly police action is primarily directed to the prevention of deterioration of life in the city as a result of increasingly disordered traffic, breakdown in communications, criminal activity. It is strictly anti-regressive.

Unless the problem of productivity imbalance between these progressive and anti-regressive activities is mastered, the previous model suggests that sooner or later the economics of a city will be dominated by the cost of the latter. And once again the underlying, psychological, cause for the resultant consequences is clearly understandable. Unless an electorate is very far-sighted, very sophisticated, the city fathers, the politicians who face re-election, the managers who seek promotion, are concerned with short-range profitability, not long-range preservation. They will in general tend to favour progressive expenditure and investment over the anti-regressive. But any such consistent bias will ultimately but inevitably lead to slowed growth and then to decay. Indeed, as we have so recently seen, only immediate chaos and decay forces the adoption of adequate levels of anti-regressive measures.

Non-uniqueness of the P and AR assignation

To avoid confusion I should perhaps stress here that the distinction between progressive and anti-regressive activity is not always clear cut.

De-pollution activity, for example, is progressive and officials can support, even encourage, it without fear of losing their office in the next election or the next shareholders meeting. The control and prevention of pollution however before its effects are felt (or, I fear, when its effects are no longer felt) is a very different matter. This is something for next year, for the next generation, to worry about. This is something for the future. The same activity is progressive when it cleans up something that has already occurred and accumulated, but is strictly anti-regressive when it is directed towards preventing the same thing from occurring.

Progressive-Anti-regressive conflict and balance

The preceding discussion can have given only the barest outline of the concepts and phenomena we are investigating. In fact we hypothesize that the life cycle of any complex, dynamic system is, at least in part, governed by the conflicting resource demands of progressive and anti-regressive activities. There will be an ever-increasing demand for the latter at a cost per unit of work that increases relative to the cost of the former.

There are three classical ways to deal with this problem. One may bring in resources from outside. In the United States, for example, an increasing number of State and City preservation projects are requesting and receiving Federal aid. But then the external source and the old system become one, enlarged, system, which too must ultimately limit, possibly decline. Alternatively one may divert resources from the progressive to the anti-regressive. This too implies a constraint on growth. A third alternative simply ignores the need for anti-regressive activity. But this is decay, even if it is not immediately visible. Thus individually or in combination these approaches all inevitably lead to a limitation to growth and hence to decay.

The way out exists. We must recognize that systems have content, structure and complexity. Consequent inertial and momentum effects result in limits on the rate of growth and of change, but not necessarily to growth itself. If one part of a system is caused to grow too fast another must grow more slowly or even decline. If at one time a system grows too fast, that must be followed by a standstill or even a decline. So the average growth rate is best restricted to its 'natural' value. Even more importantly the integrals of P and AR activity and their productivity growth must, on the average, be relatively balanced, however tempting it may be to leave the latter for future generations.

Anti-regressive activity in organizations

Before passing to the final theme of this evening's talk I would briefly mention two further points. First let me describe this same conflict as seen in the life of almost any organization. In a large business, for example, there are basically three levels of activity. The progressive rank and file engage in the activity that yields the revenue that enables the business to exist and to grow. Executive management develops and initiates the policies and strategies that enable the business to prosper. This is also progressive. Finally one has middle management. The function of middle management is to act as a communication link, vertically and horizontally. They transmit, interpret, coordinate, protect and generally prevent the development of misunderstandings, harmful competition and internal conflicts between different sub-organizations and products.

Middle management is a strictly anti-regressive, communications, responsibility. As such it will tend to grow more rapidly than the progressive sectors of the organization in both activity and unit cost. Moreover feedback analysis of the mechanisms of promotion, demotion and resignation from an organization shows that the average level of competence in the middle management ranks of an organization may be expected to decline with time [Lehman 68*]. Unless carefully and consciously controlled, middle management will tend to be a growing but increasingly mediocre organism. Thus a business, a government, even an educational institution, will get choked by its bureaucracy, the anti-regressive middle management, unless structure, function and productivity are carefully balanced. In particular it is vital that adequate resources for productivity development are correctly allocated over the organization, so that at all times productivity in the middle management area balances that of the remainder of the organization.

The Club of Rome

Let me also make very brief reference to the Club of Rome report on the Limits to Growth [Meadows 72]. In their summary (Figure 15) of the problems whose solution would help conquer the problems of growth limitation, all except one would be classified by me as anti-regressive. The fundamental oversight of the team was, perhaps, identification of resource exhaustion as the primary cause of the immediate danger. In my judgement it is, in fact, primarily a symptom. The phenomena which they discuss appear to represent just another special case of my general rule.

The problem in this instance is that mankind has, by and large, not been willing to invest sufficiently in the anti-regressive activity of long-range research and development, to find materials and techniques to replace currently conventional sources of energy and raw materials when

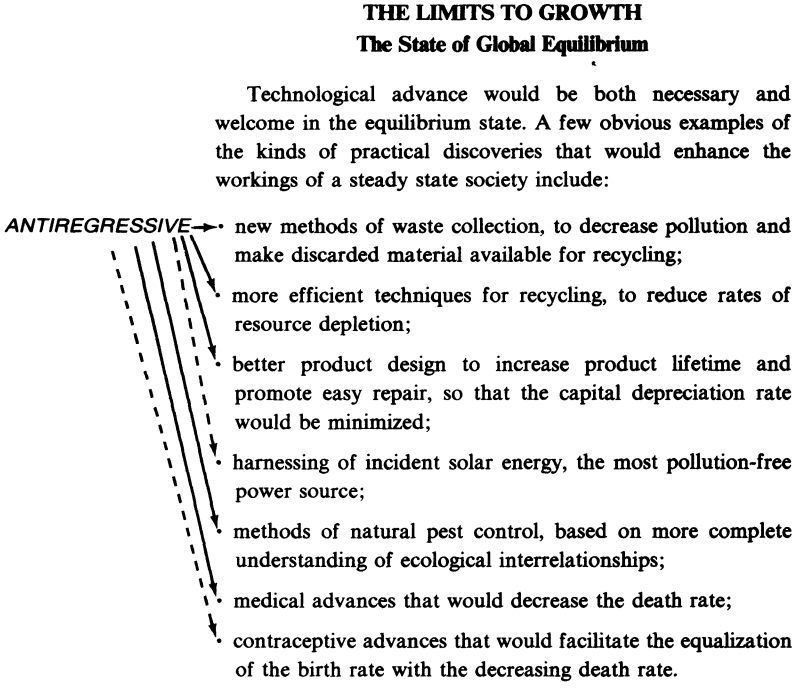


Figure 15 The true limiter of growth.

these are exhausted. The disastrous consequences of the exhaustion of specific resources is a direct consequence of the universal tendency to neglect apparently unprofitable anti-regressive activity, to prefer immediate profit to long-term security.

The sun, the atmosphere, the tides, for example, form an almost inexhaustible but as yet barely exploitable energy source. And given energy and mankind's ingenuity and creativeness, material shortages can always be overcome, provided that there is no practical limit to the growth of mankind's collective intellect. And that is the topic to which we now turn.

Limits to growth through the educational process

Knowledge or understanding

In education too there is the progressive and the anti-regressive. And, in my judgement, the same error, the analogous unbalance, exists and unfortunately increasingly so.

But let me explain. The global objectives of an educational system are rarely stated or even recognized. One may hear talk of local or immediate objectives but the really fundamental goal is not discussed. In my wanderings through the world I have experienced various educational systems, as a student, as a parent, as a manager, as a teacher. Some have stressed knowledge, others understanding. The implications of this difference reach down into the very structure and heart of the society in which these systems operate. Educational systems may, in fact, be classified by the extent of their orientation to *knowledge* and to *understanding*.

In the limit, the objectives implied by these orientations are orthogonal, at times even contradictory, in terms of the methodology they imply and the results that they yield. Orthogonality does not, of course, imply exclusion. Most systems contain, and need to contain, elements oriented to and supporting both knowledge and understanding. But what we are seeking here is correct balance between them, for the individual and for society.

Knowledge

An educational system may primarily seek to transfer to its students some selected portion of the reservoir of human knowledge, as well as the means whereby this reservoir may be further accessed. Thus the knowledge which has been accumulated over millenia will be explicitly carried forward into the future.

The drift to knowledge-oriented education occurs because its pay-off is immediate. The child comes out of kindergarten, is ready to go into school to absorb more knowledge there. It comes out of school and is able either to take a job and earn its living or to go forward to higher education and then earn its living. This is progressive. Each quantum of knowledge that is gained becomes the basis for further learning or for other activities as required by society. Moreover in this era of democracy and human equality a common standard of knowledge acts as the great leveller; it provides a definable standard against which egalitarianism can measure its success.

Understanding

At the other extreme, the pure understanding-oriented system seeks to instill insight; not so much the facts themselves as the relationships between the facts, valid analogies, cause and effect. An essential ingredient of this educational process is the development of analytic powers, self-expression and creative thinking, with the ability to seek out and elucidate new knowledge as required. Assessment of the degree of success achieved in the process is a matter of judgement, even viewpoint and opinion.

An understanding-oriented system is primarily anti-regressive. Its graduates cannot normally apply their insight and creative skills directly. Real life requires facts. But only a person with understanding can discard surplus or outdated knowledge. Only he or she can replace large depositories of knowledge by algorithms for their retrieval. It requires true understanding to develop new methodologies and to guide the knowledgeable to new horizons when the old ones become outdated or exhausted.

Attributes of the two systems

What are the attributes and hallmarks of these alternative systems? How does one determine in which direction a particular system is oriented? How can one structure a system and adopt appropriate methodologies to achieve a best balance between understanding and knowledge for the individual, for a community and for mankind at large?

The knowledge-oriented system is typified by excessive emphasis on self-learning and homework, grades and credits. It relies heavily on frequent examinations that test the extent of a student's knowledge on the completion of each quantum of study. High pass marks and high scores result from a precise definition of what was to have been learned, and from the statistical reliability of large numbers of multiple choice questions.

In the understanding-oriented system on the other hand emphasis must be quite different. Self-study and home learning start much later and the emphasis in marking will be on *how* a complex question was approached, not only on the correctness of the final solution. The award of a degree is not the sum of a large number of credits in different subjects each of which has been obtained immediately after a course has been taught. It follows from the completion of a course of study, for a demonstration of the nature and extent of the understanding that has been gained, of the meaning and the significance of what has been learnt. Multiple choice questions are of no use at all here. A student cannot express his understanding by marking a square black. His insight and ability is revealed by his choice of words, by the way in which he approaches a problem. We are interested to see *how* he has understood the facts and the concepts to which he has been exposed, what he can *do* with them, not so much in his ability to reproduce them.

Consequences of knowledge orientation

In a society in which education is knowledge-oriented the teacher adjusts his rate of teaching to the average absorptive capability of the class. Thus there will emerge a group of people clustered around a level of knowledge

determined by their average absorptive capacity. The weaker ones will have been helped forward and this indeed is a good feature of the system. Those with more powerful intellects, a greater potential for creative thinking, will have been held back. They will not have been taught to express their own ideas but merely to repeat what they have heard from their teachers.

The consequences will be a society in which achievement must be obtained through teamwork. A knowledge-oriented society will be a technology-based society in which known facts are produced from the joint encyclopaedic memories of a team and applied to achieve specified and known to be achievable objectives, great works of technology. In themselves these works do not foster creative inventiveness or the development of new concepts or deeper insights.

Such a society must possess the capability to get people to work together, the ability to manage teams of people and guide them towards a common objective. It is knowledge orientation that leads to the need for management science. Thus it is not in the least surprising that the USA with its 'melting-pot' society and, as a consequence, its knowledge-based, materialistic, orientation is the undoubted world leader in technology and in management.

And those of understanding

When we now consider the understanding-oriented system, the first thing to be noticed is that one needs more gifted teachers. Each student will interpret the teacher somewhat differently. If the teacher in turn understands the educational process, he can take each student to the limits of his individual ability. Inevitably some people will be left behind because they do not have the motivation or the ability to understand. Higher education will be more restricted and in the output of the overall educational system one finds a wide spectrum of knowledge, skills and creative ability, rather than the cluster produced by the knowledge-oriented system.

At one end of the spectum will be highly motivated, creative, thoughtful individuals able to express themselves, to develop and to exploit new concepts. They will not require to interact strongly and intimately with others to achieve their inspiration because they have learnt to achieve results on their own or in very small groups. The demand for strong, effective management, and the ability to provide it, will not be so well developed and ultimately that may prove to be to the detriment of that society. It will, for example, tend to be scientifically creative rather than technologically productive: able to produce concepts, fresh viewpoints, new inventions and methodologies, but less able to develop and control their mass exploitation.

Mankind's intellectual growth

I should not like these remarks to be in any way misunderstood. Mankind needs both systems. The challenge today is not which way to go but how to provide the correct balance between the progressive and anti-regressive in education, as elsewhere. Knowledge alone will not suffice to ensure survival of humanity. Nor can we hope to retain all the knowledge that humanity has gathered over the last five thousand years. What must be done is to ensure that forgotten knowledge can be reproduced when required. New ideas must be produced faster than the older ones may be forgotten. Understanding-oriented education is an investment in the future. It ensures that there will always exist those creative minds that can think for themselves, that produce and explore new concepts, that provide a sense of direction, creative inspiration and action, possibly even a little bit of sanity, in this fact-oriented, technology-based, profit-seeking world. But knowledge is also required. At any given moment in time only a knowledge-oriented technological society can solve the immediate problems that face a community, a nation and mankind as a whole.

We have analyzed education in terms of the progressive and anti-regressive system concepts. If the same laws of growth apply we can ensure continued growth only by achieving the correct balance between progressive and anti-regressive education. Thus I would like to think that we here at Imperial College will help counter-balance a worldwide trend to knowledge-oriented education, by maintaining and emphasizing the traditional understanding-orientation of British education. The new Computer Science course that we have set up has been carefully designed and structured to yield a student who understands and not just knows his subject. Thus I trust that we shall be playing our part in ensuring the continued intellectual growth of mankind and through that also its physical progress—progress in which computer systems and computer science will play a major role.

Conclusion

Finally may I be permitted to follow ancient Jewish practice and conclude my talk with a biblical viewpoint of the concepts I have presented. The author of *Proverbs* also recognized the potential conflict between progressive temptation and anti-regressive needs (Plate IVb):

> 'Go to the ant, thou sluggard, consider her ways and be wise. Though having no officer, overseer or ruler, she prepareth her bread in the

summer, gathereth her food in the harvest.' [Solomon, *Proverbs* 6, verses 6–8.]

One might ask why the author here places bread preparation in summer *before* food gathering in the harvest when in fact in the Middle East the harvest is gathered in spring. The Malbim, a 19th century commentator, remarks that the meaning of this text is that the ant eats in the winter because she has previously prepared her food in the summer. This she can do because during the spring harvest she is willing to store her collection rather than to eat it all. She works for a future that she may never live to see and this voluntary action occurs despite the absence of pressure from rulers or overseers. The anti-regressive is so integrated into the progressive that the two remain balanced and in step at all times.

There is clearly no need to add anything further to these ancient words of wisdom except a final overall summary:

Programs and Cities and Students
All have the potential to grow.
With P and AR action balanced
There need be no Limit to Growth.

Acknowledgments. I would like to acknowledge the loyal support, the patience and the constructive criticism that I have received from my colleagues in the Computing Science section of the Department of Computing and Control over the last two years and during the preparation of this paper. In particular I single out the many thought-provoking discussions with A. L. Lim. Equally I must mention and gratefully acknowledge all I have learned and am continuing to learn from my colleagues in the IFIP working group WG2.3. Above all, however, I want to acknowledge the close and continuing association with L. A. Belady who, but for the nature of this paper, would surely have been a co-author. Last, but certainly not least, I want to thank my wife Chava for her artistic contributions to this lecture and for her constant and continuing support and encouragement which have made my work and this lecture possible.