Int J Adv Manuf Technol (1999) 15:907–913

The International Journal of Advanced Manufacturing Technology

Configuration Design of Complex Integrated Manufacturing Systems

Y.-C. Chou

Graduate Institute of Industrial Engineering, National Taiwan University, Taipei, Taiwan

The configuration design of complex integrated manufacturing systems such as semiconductor wafer fabricaton plants is a multi-objective, multi-criterion design problem. This paper describes a qualitative reasoning methodology for the automated configuration design of manufacturing systems. The amounts, incremental qualitatives, and rates of change that make up the components of a qualitative model are first derived. They provide a formalised body of knowledge concerning the relationship between throughput, flow-time and use. A computation procedure and a system architecture that use an embedded queueing network program are then described. The methodology enhances the coherence of rule-based design knowledge, and can automatically generate a configuration design that satisfies specified performance requirements.

Keywords: Automated design; Configuration design; Qualitative reasoning

1. Introduction

Over the past several decades, the application of computer technologies in manufacturing system integration has created new large-scale systems. Flexible manufacturing systems (FMSs) are an early example of such integrated systems. Semiconductor wafer fabrication plants (i.e. fabs), comprising several hundred machines that are shared by hundreds of process steps and linked by automated material handling systems, are more recent examples. Characterised by complex material flow and alternative routeings, both types of system have several prominent design and planning issues [1]. In addition, in a dynamic business environment these systems must be continuously redesigned or modified in order to achieve specified performance goals in machine use, flow-time and work-in-process (WIP). For example, flow-time reduction is a continuous endeavour in many wafer fabs [2,3]. As the scale, complexity, and continuous nature of the design activity have increased, there is a strong need for design automation and software tools.

Implementation of automated design requires a design process model. Designing manufacturing systems could be viewed as made up of a sequence of activities or tasks. Eversheim and Herrmann advocated a planning procedure consisting of the following steps: selection of system structure; determination of machines; determination of automation level; material flow layout; and development of information and control systems [4]. Similarly, Stecke enumerated and then grouped design decisions into two phases: initial specification decisions; and subsequent implementation decisions [1]. Although the number of steps differs, they and other researchers share the perspective that can be called a task model of design. Alternatively, system design can be viewed as an iterative process of incremental refinement or solution patching [5]. Following this process perspective, the same design decisions are made at multiple levels of abstraction. These two perspectives are complementary. Following the task model, researchers have developed many decision-making tools for individual design tasks [6–10]. The process perspective, which is rooted in design automation research, is more recent and has accumulated far fewer research results.

Three design stages can be identified in manufacturing system design: conceptual design; configuration design; and detailed design. In conceptual design, product demands, processing requirements, and machine and tooling capabilities are analysed. Machine specifications, tooling concepts, material handling needs, and system structure are then determined or developed. Design evaluation is primarily guided by processing requirement- and capability-related criteria. System specifications resulting from conceptual design are typically expressed as constraints. In the configuration design stage, resources and their quantities must be selected or developed through an iterative process for each feasible configuration. Design criteria are primarily related to throughput, cycle time, WIP and tooling inventory, product quality assurance, and system flexibility. In the detailed design stage, system components are integrated and material flow, space requirements, facility layout, and other logistic, operation and scheduling issues are addressed.

A large body of models that provide aids to configuration design now exist in the literature. The most significant models

Correspondence and offprint requests to: Y.-C. Chou, Graduate Institute of Industrial Engineering, National Taiwan University, 1 Roosevelt Road, Sec. 4, Taipei, Taiwan 106. E-mail: ychou@ccms.ntu.edu.tw

are probably queueing network models that correlate material flowrates between workstations of a manufacturing system, and calculate throughput rates, product flow times, and machine use [9]. Suri and Hildebrant gave an example of using one such model in designing an FMS [11]. The configuration variables included in his model are the number of machines, pallets, and material transporters. The performance measures are concerned with throughput, use, product flow time, and waiting time. To configure a system, several iterations of analysis using the model are usually required. In each iteration, the output from the previous iteration is analysed and the system configuration is modified.

Mathematical programming techniques have also been applied to various configuration problems. Some of these models have been collected and published and there are already several reviews [7,10]. Therefore, a review of the models will not be repeated here. These models typically have discrete variables for alternative machines, tools, parts, fixtures, operations, and other equipment. The resulting formulations are usually 0–1 nonlinear programs, and the solution procedure amounts to finding a configuration in the configuration space while optimising a certain single-valued objective function [11,12].

This paper adopts a process view of design. The objective is to develop a design methodology that accommodates human reasoning as well as the design process in a design environment. The following functional requirements are stipulated for the methodology and environment:

1. They must support the resolution of multiple objectives and criteria.
2. They must support an iterative refinement process of design and human interaction.
3. They must be based on the premise that design variables and their values are too numerous to be efficiently enumerated.

In the remainder of this paper a qualitative reasoning model for the automated configuration design of integrated manufacturing systems is presented. In Section 2, the characteristics of integrated systems are discussed and the configuration design problems are defined to establish a background for this work. In Section 3, concepts of qualitative reasoning as related to configuration design are described. A configuration design methodology is then described in Section 4. The methodology has been implemented in the expert system shell language CLIPS and the C language, using an embedded queueing network model for system performance evaluation. Software system architecture for a design environment is described in Section 5.

2. Description of the Problem

An integrated manufacturing system refers to an automated production system in which programmable machines are linked by material handling equipment, and production is controlled by a near real-time shop information system. There are a number of stations, and each station contains one or more machines of the same type. Performance requirements usually include throughput, flow-time and machine use. These requirements are frequently expressed as numerical (upper or lower) bounds.

There are three aspects of the relationship between throughput, lead-time and WIP in the steady state [13]. First, the throughput of a manufacturing system increases with an increase in WIP. The increase, however, is a diminishing effect. When WIP reaches a certain level, the increase in throughput due to an increase in WIP will begin to level off. A second aspect of the integration effect is concerned with WIP and average flow-time. As WIP is increased, average flow-time will also increase. At the point where the effect of WIP on throughput levels off, the system is considered to be fully loaded. A further increase in WIP will only create more congestion, which will not significantly increase throughput but will significantly increase flow-time. The third aspect of the integration effect concerns the relationship between throughput and flow-time. An increase in throughput, if achieved by increasing the capacity of machines, will be accompanied by a decrease in flow-time. However, if an increase in throughput is achieved by increasing the WIP, flow-time will also increase. The above-mentioned relationships are highly nonlinear and are not exhaustive.

The criteria for configuration design usually interact and conflict with each other. There are also multiple means of achieving each goal. For instance, to increase production throughput, more machine resource could be added, special tooling could be developed, or process routeing could be modified to reduce the workload of the bottleneck machines. Because excessive WIP inventory and lead-time slacks are deliberately removed from integrated systems, such interaction between goals and means becomes more acute and has been characterised as the integration effect [14,15].

Configuration design is concerned with determining the type and quantity of manufacturing resources in each station, the WIP level, transport capacity, and other secondary resources. The problem of configuration design is one of finding a feasible solution in a potentially unbounded solution space such that performance requirements are satisfied.

Following the usual convention, machines, material handling equipment, and secondary resources will be called servers, transports, and tools, respectively. In addition, work-in-process inventory is also considered as a resource.

System Configuration Variables

A system configuration can be expressed as a vector of system parameters, including:

S_i the number of servers at station i, $i = 1, \ldots, M$, where station M is the transport

N the level of work-in-process, measured in pallets or cassettes

V delivery rate of a transport

Performance Measures

The parameters that describe system behaviour are:

P aggregate throughput

U_i server use at station i, $i = 1, \ldots, M-1$

U_M use of the transport
W_i waiting time at station i, $i = 1, \ldots, M-1$
W_M queue time at the transport station

Performance Requirements

These are expressed as lower and upper bounds:

$P_{\min}$ minimum throughput
$\{U_i\}_{\min}$ minimum server use at station i, $i = 1, \ldots, M-1$
$\{U_M\}_{\max}$ maximum transport use
$\{W_i\}_{\max}$ maximum waiting time at station i, $i = 1, \ldots, M-1$
$\{W_M\}_{\max}$ maximum transport waiting time

Process Data and Derived Data

w_i machine processing rate at station i
q_i probability that a part is delivered to station i by the transport

Note that flow-time of a product type can be computed from the summation of processing time, queue time and transportation time for all stations that are visited by the product. Performance requirements are specified for stations rather than for products without loss of generality.

3. Qualitative Reasoning

Qualitative reasoning seeks to formalise the common sense way of reasoning about physical systems. Its applications typically deal with physical systems with time-dependent variables governed by physical laws [16]. In this paper, qualitative reasoning is applied in the problem domain in which decision variables are not time varying but instead are related by complex relationships. This characteristic will affect the way incremental qualitatives (see below and the Appendix) are derived. Therefore, the model presented in this paper is different from those that deal with physical laws. Nevertheless, this work is based on that of Forbus [17], de Kleer [18], and de Kleer and Brown [19].

A qualitative reasoning model has three types of qualitative values that must be defined: amount, incremental qualitative, and rate of change.

1. An amount is a *qualitative value* associated with a parameter of a physical system. An amount is usually represented as an interval or partition, instead of a real number, in which the *quantitative value* of the parameter is supposed to lie. In this paper, the qualitative values of system performance measures such as throughput, transport waiting time, server use and station waiting time are represented as amounts. These amounts are either [+] or [–] in value, indicating that the performance requirements have been met or not. Both performance measures and system configuration parameters will be depicted by their variable names in brackets. For example, if the throughput P of a certain configuration is higher than the required goal, it will be given as: $[P] = [+]$.
2. An incremental qualitative is a direction of change in the value of an amount. An incremental qualitative can be either increasing (+), decreasing (–), constant, or indeterminate (?). In this paper, the direction of changes in performance values owing to changes in system configuration parameters is modelled as an incremental qualitative. The incremental qualitative of a variable is represented by the name of the variable with a prefix letter “d” and bracket enclosures. For instance, the factual knowledge that throughput will increase or decrease with the WIP level can be expressed by the incremental qualitative equality $[\mathrm{d}P] = [\mathrm{d}N]$, indicating that the direction of change in P is the same as the direction of change in N.
3. A rate of change is the qualitative value of a change that affects an amount. Rates of change are used to “quantify” incremental qualitatives by dividing possible values of a variable into intervals and then specifying what causes a performance value to be in one interval or another. For example, the knowledge that an increase in WIP level will yield a higher increase in throughput when the system is not saturated than when it is saturated can be expressed by:

   $[\mathrm{d}P/\mathrm{d}\ S_i]$ = [High], when station i is [bottleneck]
   $[\mathrm{d}P/\mathrm{d}\ S_i]$ = [Low], when station i is [not bottleneck]

   where [High] and [Low] are two defined values for the rate of change $[\mathrm{d}P/\mathrm{d}\ S_i]$.
   This qualitative reasoning model is a computation model for predicting the performance change resulting from a system configuration change. Table 1 is a calculus for this computation model. A brief example of using these three types of qualitative values and the calculus is shown in the following.

Table 1. Qualitative calculus.

[X] + [Y]		[Y] [–]	[Y] [+]	[X] * [Y]		[Y] [–]	[Y] [+]
[X]	[–]	[–]	[?]	[X]	[–]	[+]	[–]
	[+]	[?]	[+]		[+]	[–]	[+]

Example. Setting the WIP level to satisfy the goals on throughput

Assume the following knowledge base exist: (facts or rules)

1. $[P] = [-]$
2. $[\mathrm{d}P] = [\mathrm{d}N]$
3. $[\mathrm{d}P/\mathrm{d}N]$ = [Low], if system is highly saturated
4. $[\mathrm{d}P/\mathrm{d}N]$ = [Medium], if system is lightly saturated
5. $[\mathrm{d}P/\mathrm{d}N]$ = [High], if the system is not saturated

Fact 1 indicates that the goal on throughput is not yet met. From 2, it is known that throughput can be raised by increasing the level of WIP (given $[\mathrm{d}P] = [\mathrm{d}N]$, and by setting $[\mathrm{d}N] = [+]$). However, it is not certain if all increases in WIP will raise the throughput to such a level as to meet the requirement. This indetermination is verifiable by the calculation: $[P] + [\mathrm{d}P] = [-] + [+] = [?]$. Rules 3, 4 and 5 are knowledge about the rate of changes. They specify the relative rates by which throughput will increase when the WIP is increased under various levels of bottleneck use. They can be used to partially resolve this indetermination. Therefore, when the system is not saturated, it is advisable to explore the course of action of increasing WIP level. In contrast, if the system is highly saturated, it is not advisable to increase WIP as the chance of meeting the throughput goal is slight.

4. A Qualitative Reasoning Methodology

The general framework for applying qualitative reasoning to configuration design has been described in previous sessions. In this section incremental qualitatives and the rates of change will be derived. An integrated manufacturing system can be regarded as a network of queues and its performance is susceptible to analysis by queueing theory. Although queueing network models usually make simplifying assumptions, substantial knowledge about the behaviour of queueing networks has been accumulated in the literature. This knowledge should and can be used advantageously in the computation framework of qualitative reasoning.

In this work, CAN-Q, a closed queueing network model [20], is used to evaluate the performance of systems. CAN-Q assumes a central transport system, first-come-first-served discipline, infinite queue length and exponential distribution for the servers. It aggregates all work parts into one type. Performance measures P, U_i and U_M can be expressed as linear functions of the workload for the material handling station (L_M) [20]:

$$P = q_M \, V \, U_M \, S_M = q_M \, V \, L_M \tag{1}$$

$$U_i = \frac{q_i \, V \, U_M \, S_M}{w_i S_i} = \frac{q_i \, V \, L_M}{w_i S_i} \tag{2}$$

$$U_M = \frac{L_M}{S_M}$$

4.1 Incremental Qualitatives

Some incremental qualitatives can be obtained by taking the derivative of the relationship between system performance measures and system configuration parameters. For example, in Eq. (1), q_M is a constant (K) for a given problem, therefore

$$\frac{dP}{dN} = K \, V \, \frac{dL_M}{dN}$$

Since the constant K and delivery rate V are positive, [dP/dN] = [+]*[+]*[dL_M/dN]. From Table 1, [dP/dN] = [dL_M/dN]. This implies that the directions of the change in P and L_M due to a change in N are the same. There are other relationships between amounts that are generally true based on our knowledge of the behaviour of queueing networks but which cannot be analytically derived. For instance, when one machine is added to a station, the overall workload for the station will increase but the average machine use will decrease. The relation

$$[dU_i/dS_i] = [-]$$

cannot be analytically derived by taking the derivative, but it is believed to be true. This category of relationships has been verified using simulation and is also expressed as incremental qualitatives. The detailed derivation or construction of incremental qualitatives can be found in the Appendix and the results are summarised in Table 2.

4.2 Rates of Change

Among configuration variables, the number of servers has the greatest effect on system performance. In contrast, both N and V have a less significant effect on system throughput but have a significant effect on flow-time.

4.2.1 Rates of Change in Throughput and Server Use

When the goals of throughput are not met, it is logical to consider increasing the number of servers at the bottleneck station. Nevertheless, changing the number of machines at non-bottleneck stations should not be ruled out as an alternative. An argument based on the theory of constraints could be made in this regard. Non-bottleneck stations are feeding stations to the bottleneck station and increasing their capacity could lead to an increase in throughput. When machines at non-bottleneck stations have a lower cost, it may be more cost-effective to add capacity to a non-bottleneck station rather than to the bottleneck station. Therefore, in this configuration model, the effect of non-bottleneck stations as well as the bottleneck station on throughput is evaluated. For this purpose, an estimate for dP/dS_i for non-bottleneck machines is needed.

From Eqs (1) and (2), the rates of change in throughput and use will be proportional to that of the traffic load L_M. Therefore, it is convenient to compute the rates of change in L_M instead of P or U_M directly. The system configuration state can be represented by a vector $CS = (V, N, S_i$ for all $i)$. Given a configuration state CS' and a modified configuration state CS'', define the relative rate of increase in L_M as:

$$\Delta L_M = \frac{L_M(CS'') - L_M(CS')}{L_M(CS')} \tag{3}$$

The difference between CS' and CS'' is merely one more or one less machine in one of the stations. Figure 1 shows simulation results that correlate ΔL_M with the use of non-bottleneck machines. Simulation runs show that when a machine is added to a non-bottleneck station, the resultant increase in L_M is an exponential function of the use at that station. This qualitative information will be modelled using regression, i.e. $\Delta L_M = a_i U_i^{e_i}$, and used to estimate the rate of change dL_M. This will be

Table 2. Incremental qualitative ($i = 1, \ldots, M-1, i \neq k$).

	dV = [+]	dN = [+]	dS_i = [+]	dS_k = [+]	dS_M = [+]
dP	[+]	[+]	[+]	[+]	[+]
dU_i	[+]	[+]	[−]	[+]	[+]
dU_M	[−]	[+]	[+]	[+]	[−]
dW_i	[+]	[+]	[−]	[+]	[+]
dW_M	[−]	[+]	[+]	[+]	[−]

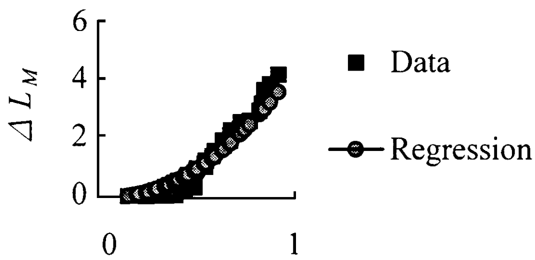

Fig. 1. Rate of change in ΔL_M.

further explained in Section 5. The rate of change for the bottleneck station is determined by extrapolation. That is, for a bottleneck station with S_i servers,

$$\Delta L_M = L_M(CS')/S_i \tag{4}$$

4.2.2 Rates of Change for Waiting Time

The rate of change in queue time at each station could be estimated using Little's law and Eq. (3). An integrated system can be modelled as a queueing network and the decomposition approach that separates the analysis of the network into node and network levels has become a standard and well-accepted technique [21]. By this approach, each station is analysed separately and characterised by the first two moments of its interarrival times at the node level. The interdependency of these two moments is then captured in the analysis of inter-station traffic flow at the network level. The ΔL_M in Eq. (3) in essence represents the increase in arrival rate to each station. From queueing network theory, it is known that the waiting time at station i is (for $M/M/c$):

$$W_i = \frac{(S_iU_i)^{S_i}p(0)}{S_i!S_iw_i(1-U_i)^2}$$

$$p(0) = \left[\frac{(S_iU_i)^{S_i}}{S_i!(1-U_i)} + \sum_{n=0}^{S_i-1}\frac{(S_iU_i)^{S_i}}{n!}\right]^{-1}$$

That is, W_i is a function of both U_i and S_i and can be expressed as $W_i(U_i,S_i)$. Assuming a constant WIP level, when a machine is added to station k, the change in W_i can be approximated using the following formula:

$$\Delta W_i = W_i((1 + \mathrm{d}L_M)U_i,S_i) - W_i(U_i,S_i) \quad \text{for } i \neq k \tag{5}$$

$$\Delta W_i = W_i((1 + \mathrm{d}L_M)U_i,S_i + 1) - W_i(U_i,S_i) \quad \text{for } i = k \tag{6}$$

4.3 Computation Processes

The configuration design process is regarded as a search process guided by qualitative reasoning. At each step of the process, there is a configuration state CS and its corresponding performance state PS. A number of candidate moves are generated and characterised based on qualitative reasoning. These candidate moves are then explored using standard search strategies of rule-based systems (see Section 5). In the following, elementary computation operations of the search process are described, followed by an example:

1. Generate a good initial configuration. An initial configuration is generated using static capacity requirements and the following constraints:

$$S_i > \frac{q_iP_{\min}}{W_i\, q_M}$$

$$\frac{\{U_M\}_{\max}\, q_iV\, S_M}{\{U_M\}_{\min}\, w_i} > S_i \;\forall i$$

2. Apply incremental qualitatives. Using qualitative calculus, those performance state variables that have a [+] value will satisfy the requirements; those variables that have a [−] value will not meet the requirement. If there are any indeterminate variables, invoke rates of change analysis.
3. Apply the rate of change analysis. If the indeterminate variable is throughput, apply the regression formula (Eqs. (3) or (4)) to obtain a prediction. If the indeterminate variable is waiting time, apply the incremental waiting time formula (Eqs. (5) or (6)).
4. Evaluate a new configuration state. The CAN-Q program is used.

In Steps 2 and 3, more than one improvement move may be identified. These moves are next classified in order to choose the more promising ones. Candidate moves are labelled with a two-letter uncertainty grade symbol. The first letter is used to indicate the confidence of performance extrapolation and the second letter will indicate whether the performance goal would be satisfied or not after the move is taken. The set of symbols and their meaning are:

CM Certainly Meet
CN Certainty Not-meet
AM Almost Meet
AN Almost Not-meet
PM Probably Meet
PN Probably Not-meet

In Step 2, if there are no indeterminate variables, the extrapolation is certain and the associated moves are labelled with the letter C, followed by an M or N. In Step 3, the extrapolation is less certain. The moves are labelled with either letter A or P, depending on whether the expected improvement exceeds 50% of the required improvement.

Example. This computation associated with a search step is outlined in Fig. 2. The initial (or given) performance state is ([+], [−], [+], [−], [+]). The action under evaluation is to add one machine to Station 1 (i.e. $\mathrm{d}S_1 = [+]$). The characterisation of this move is (CM, CN, AM, AM, PM).

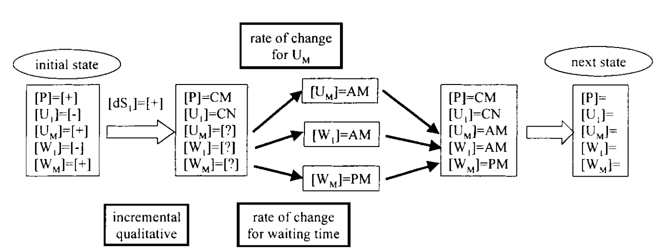

Fig. 2. An example of performance prediction.

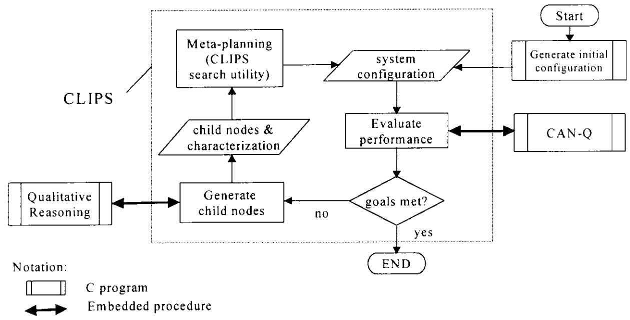

Fig. 3. System architecture.

5. System Architecture

This qualitative reasoning model of configuration design has been implemented using the expert system shell language CLIPS [22] and the C language. The system architecture is shown in Fig. 3. The core of the system is implemented in CLIPS to make use of its symbol manipulation and search strategy capability. The initial configuration generation, the CAN-Q program and the qualitative reasoning modules are implemented as embedded procedures in the C language, and can be called by the rule-based system. In the design process, each system configuration is evaluated using CAN-Q. If all performance goals and constraints are satisfied, the design process terminates. Otherwise, child nodes of the configuration are generated. The performance of each child node is estimated and characterised using qualitative reasoning logic. The characteristics are used by the meta-planner to compute a utility value to guide the search process. If after a certain number of iterations, a feasible solution is still not found, the meta-planner will conclude that the performance goal and constraints are too tight and must be relaxed. In the current implementation, the regression formula for dL_M is obtained by exploring the neighbourhood of the initial configuration. This is done by data fitting at the beginning of a design session after the initial configuration is generated.

6. Conclusion

There are unpredictable behaviour problems such as cycling of rule firing if rule-based systems are used in automated design and if the rules (i.e. the knowledge base) are not coherent. In this paper, a qualitative reasoning model is described for organising knowledge about system configuration and queueing-network related system performance measures, including throughput, flow-time, and use. A design process methodology is presented for multi-objective, multi-criteria configuration design problems using rule-based systems. The methodology enhances the coherence of rule-based design knowledge, and is able to generate automatically a satisfactory configuration design that satisfies specified performance requirements. It is a basis for automated configuration design of complex, integrated manufacturing systems such as semiconductor wafer fabs.

Acknowledgements

This work is partially funded by National Science Council of R.O.C. under NSC–87–2622–E002–011 and NSC–88–2212–E002–062.

References

1. K. E. Stecke, “Design, planning, scheduling, and control problems of flexible manufacturing systems”, Annals of Operations Research, 3, (3), pp. 51–60, 1985.
2. Friedrich Bobel and Olivier Ruelle, “Cycle time reduction program at ACL”, IEEE/SEMI Advanced Semiconductor Manufacturing Conference and Workshop, pp. 165–168, 1996.
3. Mani Janakiram, “Cycle time reduction at Motorola’s ACT Fab,” IEEE/SEMI Advanced Semiconductor Manufacturing Conference and Workshop, pp. 465–469, 1996.
4. W. Eversheim and P. Herrmann, “Recent trends in flexible automated manufacturing”, Journal of Manufacturing Systems, 1(2), pp. 139–147, 1982.
5. David G. Ullman, “A taxonomy for mechanical design”, Research in Engineering Design, 3, pp. 179–189, 1992.
6. M. Hausknecht, “Expert system for configurations planning of flexible manufacturing systems”, Proceedings, 7th International Conference on Flexible Manufacturing Systems, IPF Publication, pp. 179–190, 1988.
7. Andrew Kusiak, Flexible Manufacturing Systems: Methods and Studies, North-Holland, 1986.
8. J. W. Mellichamp, O. Kwon and A. A. Wahab, “FMS designer: An expert system for flexible manufacturing system design”, International Journal of Production Research, 28 (11), pp. 2013–2024, 1991.
9. Janes L. Snowdon and Janes C. Ammons, “A survey of queueing network packages for the analysis of manufacturing systems”, Manufacturing Review, 1(1), pp. 14–25, 1988.
10. K. E. Stecke and R. Suri (ed.), Proceedings, Third ORSA/TIMS Conference on Flexible Manufacturing Systems: Operations Research Models and Applications, Elsevier, 1989.
11. Rajan Suri and Richard R. Hildebrant, “Modeling flexible manufacturing systems using mean-value analysis”, Journal of Manufacturing Systems, 3(1), pp. 27–38, 1983.
12. Yves Dallery and Yannick Frein, “An efficient method to determine the optimal configuration of a flexible manufacturing system”, Proceedings, Second ORSA/TIMS Conference on Flexible Manufacturing Systems: Operations Research Models and Applications, pp. 269–282, August 1986.
13. Ulrich A. W. Tetzlaff, “Optimal equipment selection for flexible manufacturing systems using closed queueing network theory”, Proceedings, Third International Symposium on Robotics and Manufacturing: Research, Education and Applications, ASME, pp. 685–690, 1991.
14. Gabriel R. Bitran, William P. Peterson and Devanath Tirupati, “Some recent advances in capacity planning for discrete manufacturing systems”, Modern Production Management Systems, North Holland, pp. 201–213, 1987.
15. John E. Lenz, Flexible Manufacturing, Marcel Dekker, New York, 1988.
16. B. C. Williams and J. de Kleer, “Qualitative reasoning about physical systems: a return to roots,” Artificial Intelligence, 51, pp. 1–9, 1991.
17. K. D. Forbus, “Qualitative reasoning about physical processes,” Proceedings IJCAI-81, Vancouver, Canada, pp. 326–330, 1981.
18. J. de Kleer, “The origin and resolution of ambiguities in causal arguments”, Proceedings IJCAI-79, Tokyo, Japan, pp. 197–204, 1979.
19. J. de Kleer and J. S. Brown, “A qualitative physics based on confluences”, Qualitative Reasoning about Physical Systems, The MIT Press, Cambridge, MA, 1985.

20. J. J. Solberg, "A mathematical model of computerized manufacturing systems", Proceedings of the 4th International Conference on Production Research, Tokyo, pp. 1265–1275, August 1977.
21. D. P. Connors, Gerald E. Feigin and David D. Yao, "A queueing network model for semiconductor manufacturing", IEEE Transactions on Semiconductor Manufacturing, 9(3), pp. 412–427, 1996.
22. J. C. Giarratano, CLIPS User's Guide, NASA, Athens, January 1991.

Appendix. Derivation of Incremental Qualitatives

It should be noted that in traditional qualitative reasoning models, the independent variable is always the time, whereas in this configuration design application, the independent variables are system configuration parameters. The following incremental qualitatives are either analytically derived or construed on the basis of experience and verified via simulation. In the CAN-Q program, performance measures are related by the following equalities:

$$P = q_M * V * L_M = k * V * L_M$$
$$L_i = (q_i * V * L_M)/w_i = k_i * V * L_M$$
$$U_i = L_i/S_i$$

where k and k_i are constants.

1. Throughput (P):
   (*a*) $dP/dS_M = k * V * dL_M/dS_M$
   $dP/dS_M = [+] * dL_M/dS_M$
   Con 1: $[dP/dS_M] - [dL_M/dS_M] = [0]$
   (*b*) $dP/dS_i = [+] * dL_M/dS_i$
   Con 2: $[dP/dS_i] - [dL_M/dS_i] = [0]$
   (*c*) $dP/dV = k * \{L_M + V * dL_M/dV\} = [+] * \{[+]+[+] * dL_M/dV\} = [+] + dL_M/dV$
   Con 3: $[dP/dV] - [dL_M/dV] = [+]$
   (*d*) $dP/dN = k * V * dL_M/dN$
   Con 4: $[dP/dN] - [dL_M/dN] = [0]$
2. Server Utilisation (U_i):
   The server utilisation is $U_i = (k_i * V * L_M)/S_i$ ($i = 1,\ldots,M-1$)
   (*a*) $dU_i/dS_k = (k_i * V * dL_M/dS_k)/S_i$
   Con 5: $[dU_i/dS_k] = [dL_M/dS_k]$ $(i \neq k)$
   (*b*) $dU_i/dS_i = (k_i * V/S_i) * \{dL_M/dS_i - L_M/S_i\} = [+]*\{(dL_M/dS_i) - [+]\}$
   Con 6: $[dU_i/dS_i] - [dL_M/dS_i] = [-]$
   (*c*) $dU_i/dV = (k_i/S_i) * \{L_M + V * dL_M/dV\} = [+] * \{[+]+[+] * dL_M/dV\} = [+] + dL_M/dV$
   Con 7: $[dU_i/dV] - [dL_M/dV] = [+]$
   (*d*) $dU_i/dN = (k_i * V/S_i) * dL_M/dN$
   Con 8: $[dU_i/dN] = [dL_M/dN]$
3. Transporter Utilisation (U_M):
   $U_M = L_M/S_M$
   Con 9: $[dU_M/dS_M] - [dL_M/dS_M] = [-]$
   Con 10: $[dU_M/dS_i] = [dL_M/dS_i]$
   Con 11: $[dU_M/dV] = [dL_M/dV]$
   Con 12: $[dU_M/dN] = [dL_M/dN]$

The following confluences are construed on the basis of queueing theory but are verified by simulation.

Con 13: $[dL_M/dN] = [+]$
Con 14: $[dL_M/dS_i] = [+]$, $(i = 1,\ldots,M-1)$
Con 15: $[dL_M/dV] = [-]$
Con 16: $[dL_M/dS_M] = [+]$
Con 17: $[dW_M/dN] = [+]$
Con 18: $[dW_M/dV] = [-]$
Con 19: $[dW_M/dS_i] = [+]$, $(i = 1,\ldots,M-1)$
Con 20: $[dW_M/dS_M] = [-]$
Con 21: $[dW_i/dN] = [+]$, $(i = 1,\ldots,M-1)$
Con 22: $[dW_i/dV] = [+]$
Con 23: $[dW_i/dS_M] = [+]$
Con 24: $[dW_i/dS_i] = [-]$
Con 25: $[dW_i/dS_k] = [+]$, $(i \neq k)$

There is indeterminacy between Con 3, Con 6, and Con 9. For instance,

Con 3: $[dP/dV] - [dL_M/dV] = [+]$
Con 7: $[dU_i/dV] - [dL_M/dV] = [+]$
Con 15: $[dL_M/dV] = [-]$

From Con 3 and Con 7, $[dP/dV] = [dU_i/dV]$. However, from Con 3 and Con 15, $[dP/dV] = [+] + [dL_M/dV] = [+] + [-] = [?]$. This indeterminacy has been resolved using simulation. The results are:

Con 26: $[dP/dV] = [+]$
Con 27: $[dU_i/dS_i] = [-]$
Con 28: $[dU_M/dS_M] = [-]$