Proceedings of the 2005 Winter Simulation Conference
M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, eds.

HIGH PERFORMANCE SIMULATION IN QUASI-CONTINUOUS MANUFACTURING PLANTS

Lan Chen
Michael Pidd

Department of Management Science
Lancaster University
Lancaster LA1 4YX
U.K.

ABSTRACT

Though the quality of discrete simulation software packages is high, most are aimed at systems in which discrete objects change state as they move from work-station to work-station. This generic model is a good fit for some manufacturing, for example, much automotive production. However is not well suited to very high-speed quasi-continuous manufacturing as found in the food and drinks industry. LanSkim is a prototype PC-based package designed for these applications. It is simple to use and runs very fast but is insufficiently detailed for all purposes. Adding full detail would make the simulations run very slowly, hence we examine the use of parallel computation to allow increased level of detail. We describe LanWARPED, based on WARPED, which is a parallel simulation models of such plant.

1 INTRODUCTION

Quasi-continuous manufacturing systems are common in the food and drinks industry. Figure 1 shows their general form, which has continuous material flows as inputs and discrete final products that are output at very high speed. The continuous sections may include cookers, mixers and blenders in which materials are processed before becoming discrete units that are packed and placed into transit containers. Such plants may produce different pack sizes of the same product, different recipe variations on the same product, or a range of products and pack sizes with similar features. Though these production systems appear to be a single continuous machine, they actually consist of a sequence of machines or elements connected by conveyor systems. The conveyors can also act as buffers, thus allowing different operating speeds, at least for short periods. Since these quasi-continuous plants operate at high speed there can be considerable waste if the various elements of the plant are wrongly sized. For hygienic reasons, this waste can rarely be reused. Very few people are required to operate these production systems, which replaced flow lines that employed many more people. Sophisticated electronic control systems control the operation of the different plant sections, with the aim of recovering the large capital investment by running the plant as near to continuously as is possible.

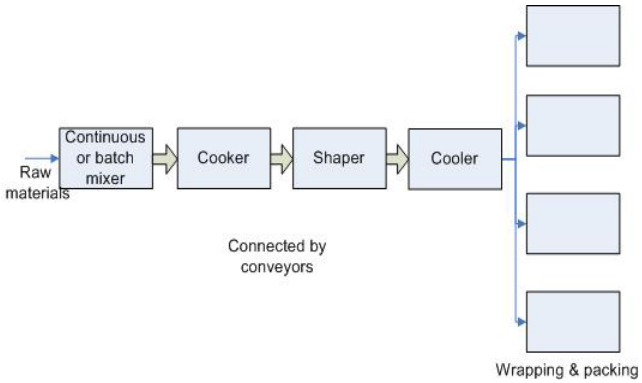

Figure 1: Schematic of Quasi-continuous Plant

Since no electro-mechanical system is ever 100% reliable, these plants typically include redundant elements such as process replicates. Thus when, say, a wrapping machine breaks down or needs its band changed, it can be stopped and another can immediately take its place. Such replicates also allow the manufacturing system to be run at varying speeds to suit different variations on the same product. Amongst other problems, designers must size the various elements of the plant to ensure it can operate efficiently with as few stoppages as possible. Thus they must decide on the speed of each element, the number of replicates and the location and size of buffers. A simulation model can be a great help in doing this.

1.1 Previous Work

Though there are many different simulation software packages on the market, the vast majority are based on a ge-

neric model that assumes that discrete objects move from work-station to work-station, changing state as they do so. Thus, many of these systems such as Witness (Lanner 2005) and Automod (Brooks Automation 2005) are widely used in the automotive industries, since these are indeed systems in which discrete objects change state. Though these simulation packages are well designed and implemented, they are not well-suited to quasi-continuous manufacturing for two reasons. Firstly, quasi-continuous plants operate at high speed (perhaps 2000 units per minute) and it may take a manufactured unit several hours for its manufacture to be complete. If this manufacturing time is, say, 3 hours, then there will be about 360,000 items to be tracked in the simulation. This can make the simulation run slowly.

Secondly, as observed earlier, quasi-continuous plants are quasi-continuous – that is, they include elements that are continuous as well as those that are discrete. This hence requires a combined discrete continuous simulation approach. Barton (1994) and Kettenis (1997) examine theoretical mathematical relationships in such combined models. Accounts of practical combined modelling include those on: pipeline construction (Shi 1997), synthetic fibre plants (Akatsuka, Furumatsu and Nishitani 1997) and pulp manufacturing tank farms (Sezgi 1994). Huda and Chung (2002) discuss combined modelling of a high-speed food processing facility using Arena. Though many discrete event simulation packages make a nod in the direction of continuous elements, these are rarely well integrated and are, in effect, add-ons. Hence, this type of manufacturing is not well served by most available simulation packages.

There seem to be 3 ways to tackle this type of problem:

1. Build greatly simplified models of the plant. In one such approach, discussed in Pidd (1987), material flow is, essentially, ignored and is treated as if it were on or off, acting as messages which control flows into and out of buffers when different plant sections stop, start and change speed. This approach does not work well when simulating high speed wrapping machines or active conveyors. Pidd (1992) describes SKIM, an easy-to-use configurable simulator developed in Pascal that takes the same approach. An alternative is to aggregate the material flows so that, instead of representing individual product items, these are grouped into rows. This approach suffers from the same problem as the first one.
2. Model just part of the plant. This is a common approach when simulating wrapping machines, since these are essentially sequential devices into which items flow one at a time. However, this ignores interactions with the rest of the plant.
3. Use high performance computation techniques to model the entire plant in some detail. The problem is that this requires significant expertise and access to appropriate computer systems. The latter is likely to be much less of a problem in future with the development of GRID computing.

If the latter route is taken, this leads (sooner or later) to a realization that parallel computation techniques may be helpful. Though these have been employed in simulating discrete parts manufacture (e.g. Lim et al. 1998 in semiconductor fabrication), there seems to have been no attempt to do so in quasi-continuous manufacturing.

Parallel Discrete Event Simulation (PDES) has been an active research area for more than 20 years and many PDES general-purpose packages have been developed, such as GTW (Das et al. 1994), WARPED (Dale et al. 1998) and SPaDES (Teo and Ng 2002). However, most of these PDES systems are general purpose and lack features to support particular applications, such as those found in manufacturing simulation (Zhang et al. 2001). It is well-known that developing PDES applications is far from straightforward and requires considerable technical expertise that may be beyond that possessed by a manufacturing engineer. Hence it seemed worth exploring the development of a specific application system that fits the domain of quasi-continuous manufacturing as suggested, in general, by Fujimoto (1993). If it could be successfully implemented it could set relatively skilled users free from the most difficult aspects of PDES so they could concentrate on the actual simulation of quasi-continuous manufacturing systems.

2 LANSKIM: A SEQUENTIAL SIMULATOR

Rather than proceed directly to the development of a PDES implementation of a generic, configurable simulator of quasi-continuous manufacturing systems, our first stage was the development of LANSKIM, a sequential simulator. This allowed the researcher (Chen) to become familiar with the research area, with the further advantage that LANSKIM can be used as a comparator with the PDES simulator. LANSKIM is based on SKIM (Pidd 1992), which pre-dates the widespread use of Windows-based computing and used the primitive graphics then available in Borland Turbo Pascal. Following SKIM, LANSKIM divides the plant elements of a quasi-continuous manufacturing system into three groups:

- Batch machines: which process a finite batch of material and must then be taken offline for a period for several possible reasons, including refilling, cleaning and recipe changes.
- Continuous machines: which receive material from a preceding process, do something to it and then make it available for the succeeding process.

- Conveyors: which are assumed to be passive and also act as buffers between plant elements. Thus, when a machine fails, there may be a delay before its downstream neighbor is starved and must therefore halt operation.

Thus, the main simulation entities are the batch and continuous machines. These take material from their input links and deliver it, possibly as discrete units, to their output link. The machines can occupy a range of states included *Stopped, BrokenDown, UnderRepair, Ready* and so on.

LANSKIM is a sequential simulator, which runs on a Windows PC, employs a three-phase simulation world view and is implemented in Visual C++. Unlike SKIM, LANSKIM is object oriented, which greatly eases the representation of different classes of machine. A simple screenshot is shown in Figure 2, in which batch and continuous machines are visible. The user selects a machine from one of these two types and then places it on a network, linking it to its neighbours. Property sheets are used to specify the parameters of each machine and the rules that govern its operation. In a quasi-continuous manufacturing plant, there are always conveyors between machines. They may be used as buffers between the operations, or there may be 'proper' buffers to cope with speed variations, in-process losses or short-term problems in the operation of the plant. So the state of material flows may affect the state of the buffer and thus affect the whole process. If a machine fails, whether it is continuous or discrete, there is a delay before the next machine is starved and this depends on the speed of the intervening conveyor. Thus the time of the starvation event is known and can be scheduled. This may, of course, need to be updated due to other changes on the plant as it runs. Thus there are continuous elements affecting discrete ones and vice-versa.

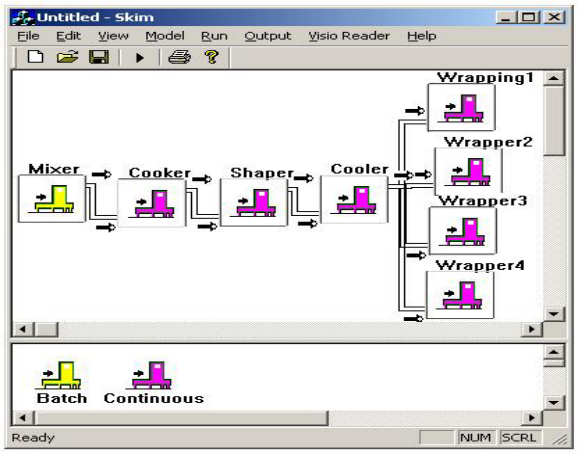

Figure 2: LANSKIM Screenshot

Three-phase simulation (Tocher 1963, Pidd 2004) is a good approach for such situations. In such a combined model the B phase can cause the execution of discrete events that might be triggered either by a machine itself (e.g. breakdowns) or by material flows (e.g. starvation or restarts) through conveyors. In the C phase, these discrete events trigger the change of the material state and then accordingly, schedule next due events.

3 A PARALLEL SIMULATOR

LANSKIM is a very simplified model of quasi-continuous manufacturing. It makes no attempt to track the movement of the hundreds of thousands of individual items likely to be active in the plant at any time. Instead, conceptually, it treats the material flow as a way to send messages from one machine to another, enabling the plant to operate as a whole system. This results in a very fast simulation, but one that ignores the important detail of the individual items. At those points in the plant at which individual items matter – for example, wrapping or filling of packs, LANSKIM is a very approximate model indeed. To model their movement requires more computer power, hence the attraction of PDES.

The terms parallel simulation and distributed simulation (Fujimoto 2000) refer to technologies that enable a simulation program to execute on a computing system containing multiple processors. In the case of distributed models, the processors are physically located in different places and connected by some form of network. The main technical problem in any PDES implementation is maintaining process synchronisation – ensuring that the future cannot affect the past – which can happen when each process is allowed to operate at its own speed. As is well known, there are two broad approaches to the maintenance of process synchronization:

- Conservative approaches: these ensure that no causality errors occur; that is that events proceed within a distributed process only if it is clearly safe to do so. The best-known example of this is CMB null message protocol (Chandy and Misra 1979, Bryant 1977), which maintains causality through null messages that promise safe computation.
- Optimistic approaches: these allow violations of local causality constraints and when they occur, roll-back the executed events to some known safe period when the processes were properly synchronised. Time Warp (Jefferson 1985) is a well-known optimistic approach.

Since speed-up is the major reason for considering PDES for this type of application, it seems sensible to use an optimistic simulation protocol (Fujimoto 1993). This has the further advantage that it is possible to add and remove simulation processes as the simulation runs and this

may be useful in such simulations. Time Warp is perhaps the best-known optimistic protocol and has the advantage of being relatively well documented (Fujimoto 2000).

3.1 Experience with WARPED

Rather than develop our own implementation of Time Warp, it seemed sensible to use available software libraries. The High Performance Cluster at Lancaster runs under Unix and supports MPI, which led us to WARPED (Dale et al. 1998), a public domain Time Warp simulation kernel, written in C++. The current release of WARPED is v1.02 and our efforts centered on this.

Implementing WARPED v1.02 on the Lancaster HPC turned out to be far from straightforward, despite the available documentation. We tried various versions of g++, including g++ 3.3, but failed to run any applications, including the game of ping-pong provided with the software. Even successful compilations, which used the correct g++ compiler and MPI-1 library, produced a host of deprecated header warnings. Following advice from the WARPED developers, we also attempted to use WARPED v2.0, but this too was unsuccessful. Hence we chose to modify WARPED v1.02, producing LanWARPED.

3.2 Modifying WARPED: LanWARPED

Warped 1.02 provides three simulation kernels:

- NoTime: an unsynchronized parallel discrete event simulation kernel,
- Sequential: a highly optimized sequential discrete-event simulation,
- TimeWarp: the optimistic parallel discrete-event simulation kernel.

Since we sought to implement a particular application suited to quasi-continuous manufacture, rather than a generic one, the NoTime kernel was of no interest. Hence only the TimeWarp, Sequential (for comparison) and Common code sets were needed. Using this source code as a base, we modified WARPED v1.02 to produce LanWARPED, which compiles and runs successfully on the Lancaster HPC, using system default implementations such as cmath.h and bool.h.

Using LanWARPED we have been able to run the default application "ping pong" and also Pidd's "Harassed Booking Clerk" (Pidd 2004). Whilst doing so, a few other problems emerged of which the most significant occurs when attempting to implement LanSkim in LanWARPED – this is the need to include quasi-continuous elements in a discrete simulation model.

The LanSkim sequential simulator discussed in section 2 uses a three-phase approach, since its C-phase provides a natural mechanism to control the switch between the continuous and discrete parts of the simulation. However, WARPED is event-based, which makes it harder to implement the switch between these two parts. To deal with this, we propose to create a commander on each LP to control the interaction between material flows and machines. The material flows are continuous and any states that occur will depend on the passage of time (e.g. a vessel is pumped dry) or on a discrete event (e.g. a machine breaks down). Likewise, some discrete events will only be scheduled if the continuous material flow achieves particular values. To avoid unnecessary repetitive computation on an LP and to make an LP switch execution between systems in an orderly manner, Commander will inform an LP which type of activities should be executed next: whether continuous or discrete. Commander instructs an LP to consider its full sequence of operations, including those on continuous parts. At the beginning of a timing loop as shown in Figure 3, the Commander instructs discrete objects to execute their discrete events that are then due. It then commands the continuous parts on the same LP to update their states according to the current simulation clock or as a consequence of discrete events. Finally, scheduled conditional events are executed.

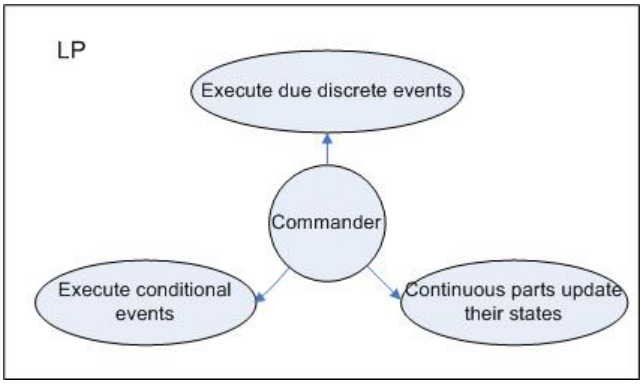

Figure 3: Commander Controlling a Timing Loop

A further problem with WARPED is its reliance on MPI-1, which forces the modeller to define the number of processes at compilation, which prevents the addition or removal of simulation processes. One attraction of Time Warp, is that processes can, in theory at least, be added and removed as the simulation proceeds. However, this is impossible under WARPED v1.02, due to its reliance on MPI-1. For the time being, our implementation of LanWARPED is similarly static.

3.3 Allocating Objects in the Physical Plant to Logical Processes

When implementing a PDES model, the allocation of 'physical' processes to logical processes is crucial. The idea is to aim for balanced computation across the proces-

sors and, as far as is possible, with limited interaction between the logical processes. High-speed quasi-continuous manufacturing plant usually takes the generic form shown earlier in Figure 1. This particular structure must be exploited if the resulting simulation is to offer reasonable speed-up. The whole production system is a sequence of processes linked by conveyors. Since LanWARPED guarantees efficient inter-process communication between objects on the same LP, we must ensure that the simulation of the plant is properly allocated to the LPs. In essence, this means that any sequence of manufacturing processes that sits in the same LP must, ideally, have only a single entry or exit link.

3.4 Implementing the Detailed Model

There are three possible versions of such a simulator that can be implemented on LanWARPED. The first is a simple transfer of the principles of LANSKIM onto the HPC via LanWARPED. This serves as a useful comparator of the complexity of such an implementation, which cannot be done by simply porting code, as LanSkim and LanWARPED are based on different simulation worldviews.

The second version is one that takes the emerging stream of discrete product units as rows that move along the conveyors together, entering the machines en route. A simple 0:1 vector can be used to hold the contents of each row, which allows it to be broken into individual product units if this is needed. This version increases the computation load as it requires, perhaps, 10,000 rows to be simulated. The third version simulates each discrete product unit as one of several hundred thousand individual entities.

4 CONCLUSION

There is a need for simulation software that can be easily used to simulated quasi-continuous manufacturing plant, since available packages rarely do this very well. LanSkim is a prototype PC-based simulator designed for that purpose. It is easy to use and runs very fast, but does not simulate enough detail to model individual product units as they move through, say, wrapping machines. Optimistic parallel simulation offers a way to achieve the speed-up that would be needed if much more detailed models were implemented. However, available open source libraries are not easy to use and seem to have implementation problems. Hence we adapted one such library to produce LanWARPED, which looks to have the power to do the job on a High Performance Computing cluster. We are currently implementing a series of models, using LanWARPED, with a view to producing a useable prototype that can be tested in an industrial setting.

ACKNOWLEDGMENTS

This work is supported by the Engineering and Physical Sciences Research Council under grant GR/R90925/01.

REFERENCES

Akatsuka, T., N. Furumastu, and H. Nishitani. 1997. Modeling and simulation of combined continuous and discrete systems: case study of spinning process. *Journal of Chemistry Engineering of Japan* 30 (5): 867-874.

Barton, P. I., and C. C. Pantelides. 1994. The modeling of combined discrete/continuous processes. *AIChE Journal* 40 (6): 966-979.

Brooks Automation Inc. 2005. AutoMod simulation software [online]. Available online via <http://www.brooks.com/pages/237_automod_simulation_software.cfm> [accessed March 16, 2005].

Bryant, R. E. 1977. Simulation of pocket communications architecture computer systems. *Technical Report MIT-LCS-TR-188*, Massachusetts Institute of Technology.

Chandy, K. M., and J. Misra. 1979. Distributed simulation: a case study in design and verification of distributed programs. IEEE *Transactions on Software Engineering*, SE-5 (5): 440-452.

Dale, E. M., J. M. Timothy, R. Radhakrishnan, and P. A. Wilsey. 1998. WARPED: a time warp parallel discrete event simulator (documentation for version 1.0). Available online via <http://www.ececs.uc.edu/~paw/warped/doc/warped.html> [accessed March 16, 2005].

Das, S., R. M. Fujimoto, K. Panesar, D. Allison, and M. Hybinette. 1994. GTW: a time warp system for shared memory multiprocessors. In *Proceedings of the 1994 Winter Simulation Conference*, ed. J. D. Tew, S. Manivannan, D. A. Sadowski, and A. F. Seila, 1332-1339. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Fujimoto, R. M. 1993. Parallel discrete event simulation: will the field survive? *ORSA Journal on Computing* 5 (3): 213-230.

Fujimoto, R. M. 2000. *Parallel and distributed simulation system.* John Wiley & Sons, Inc.

Huda, A. M., and C. A. Chung. 2002. Simulation modelling and analysis issues for high-speed combined continuous and discrete food industry manufacturing processes. *Computers and Industrial Engineering* 43 (3): 473-483.

Jefferson, D. A. 1985. Virtual time. *ACM Transactions on Programming Languages and Systems* 7 (3): 404-425.

Kettenis, D. L. 1997. An algorithm for parallel combined continuous and discrete-event simulation. *Simulation Practice and Theory* 5 (2): 167-184.

Lanner Systems. 2005. Witness [online]. Available online via <http://www.lanner.com/> [accessed March 16, 2005].

Lim, C-C., Y-H. Low, B-P. Gan, S. J. Turner, S. Jain, W. Cai, W-J. Hsu, and S-Y. Huang. 1998. A parallel discrete-event simulation of wafer fabrication processes. *Proceedings of 3rd High Performance Computing (HPC) Asia*.

Pidd, M. 1987. Simulating automated food plants. *Journal of Operational Research Society* 38 (8): 683-692.

Pidd, M. 1992. SKIM: a simulation sketchpad for plant design. *Journal of Operational Research Society* 42 (12): 1121-1133.

Pidd, M. 2004. *Computer simulation in management science(5th edition)*. John Wiley & Sons, Chichester, UK.

Sezgi, U. S., A. G. Kirkman, H. Jameel, H. Chang, J. J. Morrison, and C. A. Bianchini. 1994. Combined discrete-continuous simulation model of a rapid displacement heating tank farm. *TAPPI Journal* 77: 213-220.

Shi, J. 1997. Continuous and discrete models for construction simulation. *Proceedings of the 1997 Annual Conference of the Canadian Society for Civil Engineering* 7 (2): 155-164.

Teo, Y. M., and Y. K. Ng. 2002. SPaDES/Java: object-oriented parallel discrete-event simulation. *Proceedings of the 35th Annual Simulation Symposium*, 222-229.

Tocher, K. D. 1963. *The art of simulation*. English Universities Press, London.

Zhang, Y., W. T. Cai, and S. J. Turner. 2001. A parallel object-oriented manufacturing simulation language. *Proceedings of the fifteenth workshop on Parallel and distributed simulation*: 101-108.

AUTHOR BIOGRAPHIES

LAN CHEN is a Ph.D student in the Management Science Department of Lancaster University (United Kingdom). She received her MSc in Operational Research also from Lancaster University in 2003. Her research interests include simulation application, especially parallel simulation in manufacturing. Her e-mail address is <l.chen11@lancs.ac.uk> and her Web address is <http://www.lancs.ac.uk/postgrad/chenl11>.

MICHAEL PIDD is Professor of Management Science And Associate Dean (Research) in Lancaster University Management School. He is known for his work in two areas: computer simulation and the complementary use of soft and hard OR. His text 'Computer simulation in management science' is now in its 5th edition and his books on complementarity include 'Tools for thinking' (in its second edition) and 'Systems modeling: theory and practice'. All are published by John Wiley. For the whole of 2004 he is a Research Fellow in the UK's Advanced Institute of Management Research, funded by the ESRC, examining performance measure in the public sector. Email to <M.Pidd@lancaster.ac.uk>, website at <http://www.lancs.ac.uk/staff/smamp>.