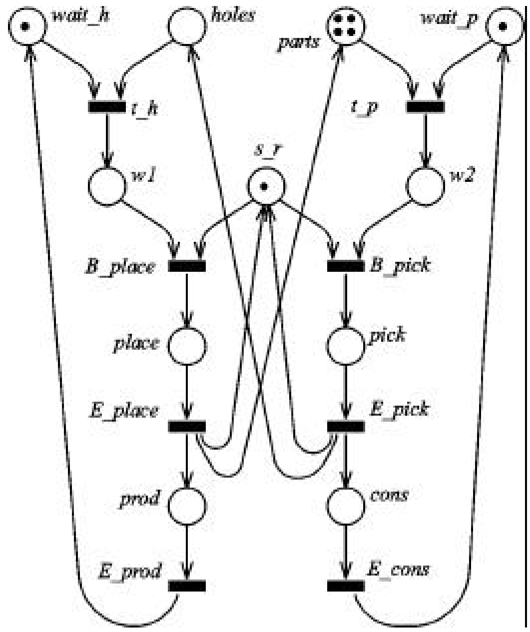
Petri nets

Petri Nets (PN's) is a formal and graphical appealing language which is appropriate for modeling systems with concurrency. PN's has been under development since the beginning of the 60'ies, when Carl Adam Petri defined the language. It was the first time a general theory for discrete parallel systems was formulated.

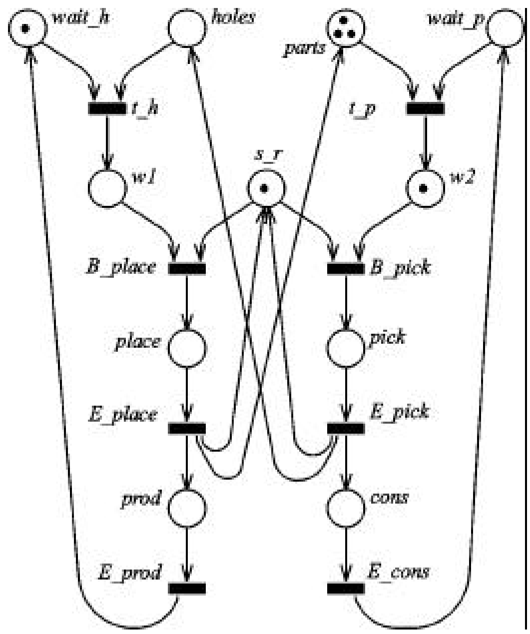
Description of a PN model. The language is a generalization of automata theory such that the concept of concurrently occurring events can be expressed. PN model is oriented to the description of both states of a system and actions producing evolution through the states. In this sense, it differs from other formal models of concurrent systems which usually are state-based or action-based. PN's treat states and actions on equal footing. In fact, the structure of a PN can be seen as a bipartite graph whose two different kind of nodes, places and transitions, correspond with states and actions of the system. Certain similarity with queueing models can be observed at this point. Storage rooms and service stations of queueing networks represent also states and actions, respectively. In queueing models the state of the system is represented by means of a given distribution of customers at storage rooms (queues). In an analogous way, a marking or distribution of tokens (marks) over the places of the PN defines the state of the system. Therefore, as for queueing networks, the representation of a state is distributed (see Figure 1.a).

The behavior of a queueing network is governed by the departures of customers from stations, after finishing service, and the movement towards other storage rooms. The token game is the analogue in PN models. Tokens are stored at places and the firing of a transition produces a change of the distribution of tokens or new marking (see Figure 1.b).

Adequacy of the paradigm. The first main property of PN models for the description of concurrent systems is its simplicity. A very few and simple mathematical entities are necessary for the formal definition of nets. This fact constitute a great advantage, mainly in the modeling of concurrent systems which are enough complicated *per se*. In spite of the simplicity of the model, its generality must be remarked. The three basic schemes in the modeling of concurrent systems can be included in the PN structure: sequencing, choice, and concurrency. Moreover, other typical and well-known elements in the modeling of distributed systems, as *rendez-vous*, shared resources, fork-joins..., can be easily derived by combination of the basic schemes.



(a)



(b)

Fig. 1. (a) An example of a PN. Places are drawn with circles while transitions are drawn with bars. The initial marking of the PN has one token in place "wait_h", four tokens in "parts", one token in "wait_p", and one token in "s_r". Transition "t_p" is enabled (it can be fired).
(b) After the firing of transition "t_p", this new marking is obtained (one token is removed from "parts", another one from "wait_p", and one token is added to place "w2"). Now, transition "B_pick" is enabled.

One aspect of the adequacy of PN models is their possibility of expressing all basic semantics of concurrency, interleaving, step, and partial order semantics, which can be compared within the PN formalism. In this sense, PN's are capable of modeling "true concurrency". Locality of states and actions constitutes another aspect of adequacy for the modeling of concurrent systems. It provides the possibility of progressive modeling by using stepwise refinements (top-down) or modular composition (bottom-up modeling).

Validation of qualitative properties. As in the case of queueing network models, the graphical representation of PN's is crucial for the interest of systems designers in this model. However, distributed and concurrent systems are complex and difficult to master for designers by nature. Therefore, desirable "good properties" must be formally defined and the model must be validated for these properties. In this sense, qualitative analysis of PN's is important before going on the implementation. A wide range of techniques for checking synchronic (lead, distance, places bounds, places mutual exclusions...) and activity properties (deadlock-freeness, liveness, home states...) are reasonably known.

Reachability analysis, based on the construction of the state space of the model, provides a complete knowledge of all its properties if the net is bounded (i.e., if the number of reachable states is finite). However, the exponential temporal and spatial computational complexity originated from the state explosion reduces the applicability of this enumeration technique.

In order to avoid the state explosion, reduction/transformation and structural techniques have been developed. The first are based on the application of local rules for the simplification of nets, preserving some of the desirable properties. On the other hand, structural techniques allow to conclude about some properties of the model just from the net structure and using mathematical tools taken from graph theory, linear algebra, convex geometry, or linear programming.

Performance evaluation of timed PN's. Regarding quantitative analysis of PN's with timing interpretation, the most commonly used technique consists on the derivation of exact performance measures from the reachability graph of the model (if bounded) which is identified with a Markov chain, under certain assumptions on the stochastic specification. As in the case of qualitative reachability analysis, the explosion of the computational complexity is the main problem in the actual use of this technique for the performance evaluation of large models. Alternative methods for the quantitative evaluation of PN models have been tried out. As in the case of queueing networks, approximation techniques and the computation of bounds constitute an option instead of exact analysis.

Application domains. Petri nets have been applied mainly as a modeling tool for the design, validation, and evaluation of repetitive automated manufacturing systems, parallel and distributed computer systems, parallel software, and telecommunication networks.

Consider, for instance, the problem of modeling and evaluating a producer-consumer system composed by two machines and a buffer storage, as depicted in Figure 2. The machine M1 produces parts that are placed at the buffer storage. The maximum capacity of the buffer is four parts. The machine M2 picks parts from the buffer for processing them. The control system for the production and consumption of parts is depicted in Figure 1.a by means of a PN. Machines M1 and M2 cannot operate simultaneously with the buffer, i.e., the pick and place operations are in mutual exclusion (modeled with place "s_r" in the PN). The left hand side of the PN (places "wait_h", "w1", "place", and "prod") represents the different states of machine M1, while the right hand side (places "wait_p", "w2", "pick", and "cons") models the states of machine M2. Places "holes" and "parts" represent the state of the buffer. The initial marking of four tokens in place "parts" is the maximum capacity of the buffer.

An example of outcome of the qualitative analysis of the PN depicted in Figure 1.a is that, for all reachable markings, i.e., for all states that the system can reach, the following invariant property holds: *the sum of tokens in places "s_r", "place", "prod", "pick", and "cons" is always equal to one*. It represents a mathematical expression of the fact that the machines M1 and M2 cannot operate simultaneously with the buffer.

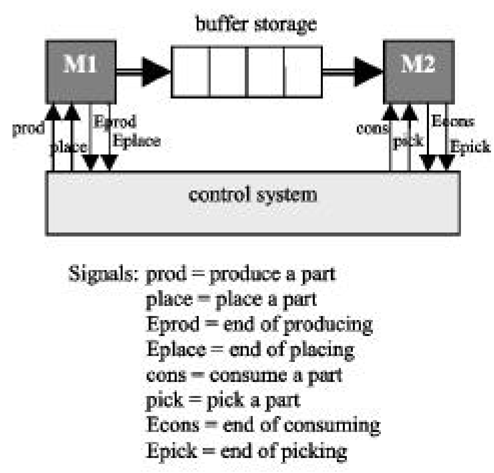


Fig. 2. A producer-consumer manufacturing system with two machines and a buffer storage. Its PN model is depicted in Figure 1.a.

Concerning quantitative analysis, also called performance evaluation of the model, a time duration can be associated with the transitions of the PN (that represent the activities performed in the system). Assuming, for instance, that the duration of activities represented by transitions "t_h", "t_p", "B_place", and "B_pick" is zero (i.e., they occur immediately), while the duration of activities represented by transitions "E_place", "E_prod", "E_pick", and "E_cons" is an exponentially distributed random variable with average 2, 4, 3, and 2 units of time, respectively, the "mean cycle time" of the system can be computed by solving a continuous time Markov chain. That mean cycle time represents the average time needed for a complete production of each part, and it is an example of responsiveness performance index of the system. Figure 3 depicts the value of the mean cycle time of the system obtained for different capacities of the buffer storage. Clearly, the response time decreases by increasing the capacity of the buffer. Nevertheless, in this case it makes no sense to dimension the size of the buffer in more than 4 or 5 units because for larger capacities the improvement of the response time is meaningless.

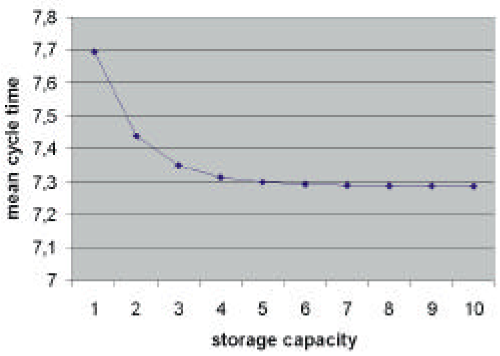


Fig. 3. The effect that the size of the buffer has on the mean cycle time of the system, for the PN depicted in Figure 1.a.

For background information SEE *AUTOMATA THEORY*, *QUEUEING NETWORKS*, *MANUFACTURING SYSTEMS*, *PARALLEL COMPUTERS*, *TELECOMMUNICATION NETWORKS* in the McGraw-Hill Encyclopedia of Science & Technology.

Javier Campos

Bibliography. J. L. Peterson, *Petri Net Theory and the Modeling of Systems*, Prentice-Hall, 1981; W. Reisig, *Petri Nets, An Introduction*, Springer Verlag, 1985; K. Jensen, *Coloured Petri Nets, Volume 1: Basic Concepts*, Springer-Verlag, 1992; K. Jensen, *Coloured Petri Nets, Volume 2: Analysis Methods*, Springer-Verlag, 1994; M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, G. Franceschinis, *Modelling with Generalized Stochastic Petri Nets*, John Wiley and Sons, 1994; F. DiCesare, G. Harhalakis, J.M. Proth, M. Silva, and F.B. Vernadat, *Practice of Petri Nets in Manufacturing*, Chapman and Hall, 1993.

URL's. University of Aarhus, Denmark, *World of Petri Nets*, http://www.daimi.au.dk/PetriNets/; University of Hamburg, Germany, *The Petri Nets Bibliography*, http://www.informatik.uni-hamburg.de/TGI/pnbib/.