Fundamental Properties of Colored Petri nets

- **Graphical programming language** which incorporates process/object concurrency.

- **Colors** are Data Types.

- Other important concepts:

- **variables** - represent bindings of identifiers to specific colored tokens

- **arc inscriptions** - represent logical and time-dependent expressions

- **code segments** - to represent elementary/complex activities attached to transitions

- **guards** - represent conditions under which activities can be performed.

Scalability of the model:

- when the size of the modeled system grows then the size of the corresponding model should remain constant
- when the size of the modeled system grows then the modeling time should grow as little as possible.

- Do Colored Petri Nets support **scalability of the model?**:

- Colored Petri nets are scalable in relation to the model size
- Colored Petri nets are not scalable in relation to the modeling time.

- why Colored Petri nets are **scalable** with respect to the model size:

- **colored tokens** permit to fold a Petri net for similar problems of increasing sizes

- **multiple transitions** can be folded (transitions that represent identical or similar system components can be folded into one transition whose different instances are enabled by tokens assigned different color values)

- **arc expressions** result in different values; during an execution the weights of arcs vary allowing a single piece of net to treat different but related (by their color set) tokens in different ways

- **parametrization of color sets** permits to adjust the same Colored Petri net structure to model related systems with different number of components.

• Colored Petri nets are **not scalable** with respect to the modeling time because:

- when modeling time includes also simulation time it is obvious that more **simulating time** is required for larger model (result of the state explosion phenomenon for parallel/distributed systems)

- **code segments** require much more time to prepare for larger systems than for a smaller one

- multiple transitions, arc expressions, transition guards, and the corresponding expressions are more complex and require more modeling time.

• **what are Colored Petri nets for?**

- **foundations of Colored Petri nets** are powerful enough to support different stages of parallel/distributed system design:

- **formal specification**
- **rapid prototyping**
- **validation**
- **performance evaluation**
- Petri nets in general, and Colored Petri nets in particular, are especially helpful for project engineers and project managers in the area of **embedded computer systems**.

Positive features of Colored Petri nets

- **design modularity** and **hierarchical behavior** (should be treated as orthogonal features)
- explicit representation of **concurrency**
- **formal analysis** capabilities
- **shorter development cycle**
- easier **maintenance**
- suitable for **architectural analysis** to modify model structure and network connectivity
- **reconfiguration** of functional activities
- easy **change of communication links** between key system components.

Modeling goals when using Petri nets:

- presentable specification
- accuracy.

What are main problems with Petri Nets?

- not good for **goal oriented problems,** i.e. optimization problems are not easily representable by Petri Nets (one could argue that some optimizations based on net theory are possible, for instance performance evaluations techniques; however, in classical sense of optimization theory, Petri nets do not incorporate well this feature).

- for realistic practical problems even using Colored Petri nets leads to the **explosion of the number of bindings**;

Example: Take an example of a single transition which has as a set of preconditions of three places with 30, 20, 40 tokens, respectively being of different colors. This situation leads to 24,000 possible bindings.

To reduce a number of bindings for simulation reasons (at least in initial stages of prototyping) one could consider a relation of

'equivalent bindings' for given transition and perform simulation for non-equivalent bindings. Definition of equivalence relation for bindings can be example-dependent.

In Fig.1 a binding with single token selected arbitrarily from each place will be sufficient to fire transition assuming that arc inscriptions x, y, z and guard g(x, y, z) are satisfied. Bindings should be considered as equivalent with respect to specific transition.

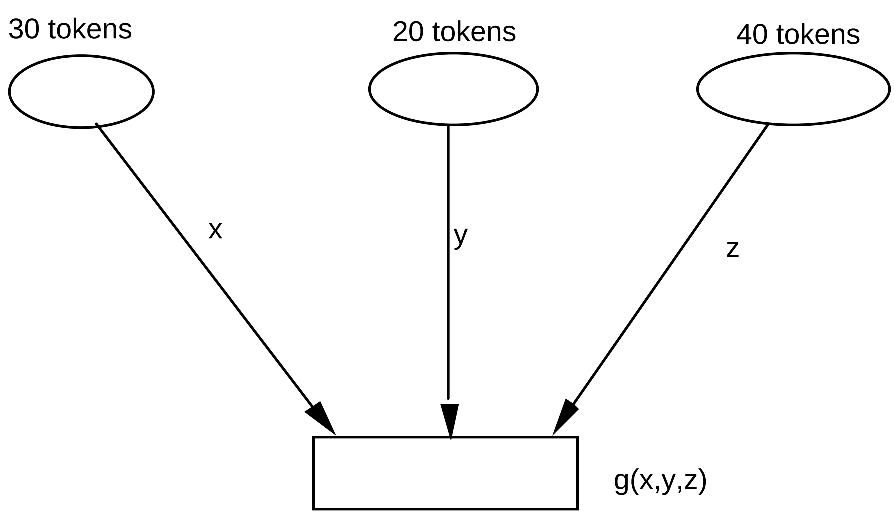

Fig.1. Example of CP-net with 24, 000 different bindings.

- The following **modeling enhancements** to the fundamental Colored Petri nets model would be very helpful for reporting and performance evaluation reasons:

- statistical analysis tools

- graphical display of performance results
- animation tools
- batch simulation run methodology.

- **functional expressiveness** of behavior only (difficulties with modeling of procedural-type activities).

- **data structures and data conventions** are too restrictive; this basically follows from usage of CPN ML language for data descriptions.

- Colored Petri nets are **not standardized**, i.e. modeling of new systems cannot take advantage of model/software reusability; this feature is sometimes called as **the lack of regularity**.

- Petri nets missed a way of **abstraction** from internal actions; i.e. **processes are not treated as "black boxes"** where only the communication behavior is important.
- **problem of compositionality,** i.e. lack of convenient way of composing and decomposing larger nets from or into smaller ones by means of high level composition/decomposition operators

- **compositionality and abstraction** are necessary to manage with the behavioral complexity of processes

- **compilation times** are too large; swapping between editor and simulator is too time-consuming

- **lack of flexibility** concerning the implementation of sophisticated numerical algorithms

- Colored Petri nets should be equipped with some other mechanism different than multi-sets because multi-sets do not provide **fairness**.