A Linear Specification Language for Petri nets

Carolyn Brown Doug Gurr Valeria de Paiva
University of Aarhus[*] University of Cambridge

October 1991

Abstract

This paper defines a category **GNet** with object set all Petri nets. A morphism in **GNet** from a net N to a net N$'$ gives a precise way of simulating every evolution of N by an evolution of N$'$. We exhibit a morphism from a simple message hanker to one with error-correction, showing that the more refined message handler can simulate any behaviour of its simple counterpart. The existence of such a morphism proves the correctness of the refinement.

Earlier work [Bro90, BG90, BG] defined a modular theory of elementary Petri nets based on de Paiva's Dialectica categorical models of linear logic. We here modify her construction, defining categories $\mathbf{M}_N\mathbf{C}$ which model intutionistic linear logic [GL87]. **GNet** arises naturally from $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$ inheriting the structure which models linear logic. This more general framework has several advantages over our previous one. The theory is simplified, we obtain precise results about morphisms as simulations, relating them to CCS, and we obtain a natural extension to marked nets.

The linear connectives are modelled in GNet by net combinators. Being functorial, these combinators op_i are such that, if each N_j is refined by a net N'_j, then $op_i(\mathrm{N}_0, \ldots, \mathrm{N}_m)$ is refined by $op_i(\mathrm{N}'_0, \ldots, \mathrm{N}'_m)$. We show that the operation of restriction also has this property, and thus (in the language of algebraic specification) our notion of refinement composes horizontally with respect to the linear connectives and

[*]Both C. Brown and D. Gum are supported by BRA CEDISYS

restriction. Furthermore, our notion of refinement composes vertically because it corresponds to categorical morphisms. These properties of our notion of refinement are precisely those required to develop an algebra of nets in which complex nets can be bit from smaller components, and refined in a modular and compositional way. We illustrate our approach with an extended example, analogous to Milner's *Jobshop* example.

1 Introduction

Petri nets [Rei85] are a long established and relatively successful model of concurrent systems. They admit an appealing operational interpretation, the "Token Game", and their simple graphical presentation makes them a convenient system with which to describe or specify concurrent systems. This is evidenced by the successful industrial application of systems such as Jensen's Petri Net Tool [Jen90]. Finally, they provide a framework in which to investigate the issues relating to non-interleaving models of concurrency.

Unfortunately, there are significant difficulties in using Petri nets to describe and specify concurrent systems, primarily the lack of good notions of refinement and modularity. By refinement we mean the process of building an increasingly detailed description of a system by progressively replacing simple components with more complex ones. Intuitively, a net N' refines N if N' incorporates more design decisions than N: in this case we expect N' to be able to simulate every evolution of N, in the sense that every evolution of N induces a corresponding evolution in N'. Modularity allows us to build complex systems from simpler component subsystems. These ideas, which correspond respectively to the notions of vertical and horizontal composition of refinements in the field of algebraic specification [Wir71, ST88], are essential to developing an implementable theory of specifications (in the manner of *Z* [AS79], *Clear* [BG80] and *VDM* [Jon86]). We expect suitable notions of refinement and modularity to satisfy certain properties. Firstly, we expect the identity operation to be a refinement. Secondly, if we have a refinement of a net N_0 by N_1 and a refinement of the net N_1 by N_2 then it should be possible to compose these to obtain a refinement of N_0 by N_2. Thirdly, we would like a number of net building operations op_i with which to construct a complex net $op_i(\mathrm{N}_0, \ldots, \mathrm{N}_n)$ from any component nets $\mathrm{N}_0, \ldots, \mathrm{N}_n$. Finally, we want our refinement to compose horizontally with respect to each of these operations op_i [ST88]. That is, if each N_j is refined by a net N'_j then $op_i(\mathrm{N}_0, \ldots, \mathrm{N}_n)$ should be refined by $op_i(\mathrm{N}'_0, \ldots, \mathrm{N}'_n)$.

Winskel [Win84] has addressed these issues by defining a category in which the objects were Petri nets and the morphisms represented partial simulations. There are several advantages to a categorical approach. Using morphisms to represent refinements ensures that we have compositionality of refinements, and furthermore that the composition of refinements is associative. That is, if we have refinements $f : \mathrm{N}_0 \to \mathrm{N}_1$, $g : \mathrm{N}_1 \to \mathrm{N}_2$ and $h : \mathrm{N}_2 \to \mathrm{N}_3$ then refining N_0 to N_2 by fg and then refining by h is equivalent to refining N_0 to N_1 by f and then refining by gh. In addition, any structure that the category of nets possesses (for example, products and co-products) yields the constructors on nets we require for a modular approach. The functoriality of the categorical constructions ensures that refinement composes horizontally with these constructors. Thus we have a basis for an algebraic theory of specification using Petri nets.

Further, categories with sufficient structure are endowed with an associated logic. Thus Cartesian closed categories correspond to simply typed lambda calculi [LS86], toposes correspond to intuitionistic logic [Fou80, Joh77] and symmetric monoidal closed categories with certain other structure appear to correspond to Girard's Linear Logic [Amb91, See89]. Therefore, if our category of nets has the appropriate structure, we obtain a logic for reasoning about nets and refinement. In addition, expressing models of concurrency as categories enables us to explore the relationships between models by exhibiting functors between the categories. Notably, Winskel and Nielson [Win84, NW91] have shown that many different models of concurrency can be related by reflections or coreflections between the associated categories. Finally, the level of generality offered by a categorical approach often makes it relatively straightforward to modify the structures under consideration.

We have described a number of advantages of the categorical approach. However, these pleasant algebraic properties are to no avail if the categorical structures do not have a meaningful computational interpretation. In particular, the definition of morphism between nets should accord with the intuitive notion of refinement, and the categorical constructions on nets should correspond to useful net–building operations. The difficulties in defining a suitable notion of net morphism are demonstrated by the large number of categories of nets which has been proposed [Bro90, BG90, BG, DMM89, MM88a, MOM89, NRT90, Win87, Win88]. Many of these categories arise as instances or as subcategories of instances of the construction we now present.

In this paper, we define a new category of Petri nets based on de Paiva's categorical models of Girard's linear logic [Gir87]. The morphisms arising from this abstract approach have an appealing computational interpretation in terms of simulations. Further, for a wide class of nets, all simulations are captured by our morphisms. Thus we obtain a compositional notion of net refinement. Our category is a sound model of linear logic and so has a rich categorical structure. In particular, it is symmetric monoidal closed and has finite products and coproducts. These constructions, together with restriction (net containment) constitute the operations we require to give a modular theory of nets. Our notion of refinement composes horizontally with respect to each of these operations. We here apply this theory in a detailed example along the lines of Milner's *Jobshop* [Mil89]. Also, we obtain a natural extension from unmarked nets to marked nets. This preserves the categorical structure required for a modular theory of net specification, and retains the interpretation of morphisms as simulation. Since our category is a model of linear logic, we can develop a linear proof system for reasoning about net refinement.

This paper contains three important extensions of our previous work [Bro90, BG90, BG]. We extend our construction from elementary Petri nets (nets in which no arc has weight greater than 1) to all Petri nets. We obtain results giving a precise understanding of the morphisms as simulations which are closely analogous to simulation in labelled transition systems, and we give a more detailed analysis of the ideas of simulation and modularity. Our model encompasses several others. Further, it admits a natural extension to marked nets which preserves the categorical structure required for a modular theory of net specification, and retains the interpretation of morphisms as simulation.

2 Summary of the Paper

In this paper we address the issue of modular specification of concurrent processes by constructing categories of Petri nets. In Section 3 we review the definitions and properties of Petri nets. Recall [DP89, DP88] that de

Paiva constructs a class of categories **GC**, in which objects are relations in a category **C** and the morphisms give a notion of map between relations. In [Bro90, BG90, BG] we constructed a category of elementary Petri nets based on **GC**. In Section 4 we modify the construction of **GC** to obtain a class of categories $\mathbf{M}_{\mathbb{N}}\mathbf{C}$ whose objects are generalised relations. In Section 5 we show that the categories $\mathbf{M}_{\mathbb{N}}\mathbf{C}$, like the categories **GC**, have sufficient structure to model Girard's linear logic [Gir87].

In Section 6 we construct a category **GNet** of general nets, based on the category $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$. In our earlier work we indicated how the morphisms could be understood as a notion of refinement or simulation. In Section 6 we make this precise. We show that whenever there is a morphism $\langle f, F\rangle$ from N to N$'$ in **GNet**, if N can evolve under a sequence of events $e_0, e_1, \ldots, e_n$ then N$'$ can evolve under the sequence of events $f(e_0), f(e_1), \ldots, f(e_n)$. We also relate this to the notion of simulation in labelled transition systems such as CCS. Finally, we give a practical example of a morphism which shows how a simple message handler is simulated by a more sophisticated one.

In Section 7 we characterise **GNet** as a limit in **Cat**. As a consequence, all the structure of $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$ lifts to **GNet**, which has finite products, finite coproducts and is symmetric monoidal closed. Thus **GNet** has sufficient structure to model intuitionistic linear logic. We study in detail the product and coproduct of two nets. In particular, we show how to represent both the synchronous product of two nets and a restricted product which allows specified asynchronous events. We prove that the behaviour of the product of two nets is the product of the behaviours of its component nets.

In Section 8 we develop our compositional theory of nets. We first give a simple condition on restrictions of a product net which ensures that refinement composes horizontally with restriction. We illustrate our theory using a detailed worked example related to Milner's Jobshop [Mil89], which builds a large net in a modular way from smaller component nets. We illustrate the horizontal composition of our refinement with respect to our constructors in the following way. We refine the "Jobber" component to introduce a distinction between hard and easy jobs, and show that there is a morphism from the old jobber to the new one and hence that the new jobber simulates the old jobber. The naturality of our constructions ensures that there is a morphism from the the old compound net to the new compound net, and hence that the new compound net simulates the old one.

In Section 9, we sketch the application of linear logic proof terms to reasoning about net refinements. This is work in progress.

In the main part of this paper, as in [Bro90, BG90, BG], we have considered nets without initial markings. In Section 10 we show how the results of Section 6 enable us to extend naturally to a category **MNet** of marked nets. There is a forgetful functor $\mathcal{U}$ from **MNet** to **GNet** which has both left and right adjoints. Thus $\mathcal{U}$ preserves any small lists and colimits that exist in **MNet**. We prove that **MNet**, has finite products and finite coproducts, describe the product and coproduct of two marked nets and discuss the monoidal closed structure.

3 Preliminary Definitions for Petri Nets

Petri nets model processes by indicating the changes in local states (conditions) which are induced by the occurrence of events. The causal dependency between conditions and events is expressed using two multirelations: the pre–condition relation indicates which conditions must be satisfied before an event can occur, while the post–condition relation indicates the conditions resulting from the occurrence of an event. These two multirelations determine the dynamic behaviour of a net.

An introduction to Petri nets is given in [Rei85]. In defining Petri nets and their behaviour we assume standard definitions concerning multirelations and multisets [Win88].

Definition 3.1 *A* Petri Net *is a 4-tuple* $\langle E, B, pre, post\rangle$, *where* E *and* B *are sets, and* pre *and* $post$ *are functions from* $E \times B$ *to* $\mathbb{N}$ *(multirelations).*

We shall call elements of E *events* and elements of B *conditions*. We shall call *pre* and *post* the *pre-* and *post-condition* relations of N respectively. We write N for the Petri net $\langle E, B, pre, post\rangle$, N_0 for the net $\langle E_0, B_0, pre_0, post_0\rangle$ and so on. We write **Petri** for the class of Petri nets. A net N is *elementary* if the images of both *pre* and *post* are contained in **2**.

With each of the multirelations *pre* and *post*: $E \times B \to \mathbb{N}$, we associate a function with the same name, from E to multisets over B, defined by the formal sums

$$pre(e) = \sum_{b \in B} pre\langle e, b\rangle b \quad \text{and} \quad post(e) = \sum_{b \in B} post\langle e, b\rangle b.$$

We call $pre(e)$ the *pre-condition set* of e, and $post(e)$ the *post-condition set* of e. We extend the function *pre* (and similarly *post*) to a multiset of events $A : E \to \mathbb{N}$ as follows:

$$pre(A) = \sum_{e \in E} A(e) pre(e) \quad \text{for any multiset } A \text{ over } E$$

Definition 3.2 *Let* N *be a Petri net. A* marking *of* N *is a function* $M : B \to \mathbb{N}$ *(a multiset over B). We write* $Mark(\mathrm{N})$ *for the set of all markings of* N*. A* marked net *is a pair* $\langle \mathrm{N}, M\rangle$*, where* N *is a Petri net and* M *is a marking of* N.

Remark 3.3 *Various authors add further conditions to the definition of a net, or of its markings, to ensure convergence of the relevant formal sums. It suffices to require that every event of the net has unite pre- and post-condition set, and that every condition is in the post-condition set of finitely many events.*

There is a graphical representation of Petri nets in which events are represented by labelled boxes, conditions by labelled circles, and the pre- and post-condition relations by weighted, directed arcs. We shall omit weights of value 1.

3.1 The Evolution of Petri Nets

We call the dynamic behaviour of a Petri net its evolution. The evolution of a net N reflects the causal dependencies of the process which N models, since the pre- and post-condition relations express causal dependency. Events which are causally independent may occur concurrently.

Let $\langle \mathrm{N}, M\rangle$ be a marked Petri net and let A be a multiset over E. We say $\rangle \mathrm{N}, M\langle$ *enables* A, written $M \downarrow^{\mathrm{N}} A$, if for each $b \in B$, $\sum_{e \in E} A(e) pre\rangle e, b\langle \leq M(b)$. Further, we say that N *one-step evolves* under A from the marking M to the marking M', written $M \leadsto_1 M'$, if

$$M' = (M - pre(A)) + post(A).$$

In this case, the events of A are said to occur *concurrently*, in what we shall call a *transition*. The *derivability relation* of a net N, written $\rightsquigarrow$, is the transitive closure of $\rightsquigarrow_1$. We say a net N *evolves* from a marking M to a marking M' if $M \rightsquigarrow M'$. We sometimes label such an evolution, writing $M \overset{A}{\rightsquigarrow} M'$.

4 A Category of Multirelations

We now extend the treatment of relations in de Paiva's dialectica category **GC** to multirelations, using a variation of Chu's construction [Bar79]. Our construction relates closely to the categories $\mathbf{GAME}_K$ [LS91] and **DecGC** [DP89].

Lemma 4.1 Let $\mathbf{C}$ be a concrete[1] category with finite products. Let N be an object of $\mathbf{C}$, equipped with a partial order $\leq$. The following data:

- objects are triples $\langle E, B, \alpha\rangle$, where E and B are objects of $\mathbf{C}$ and $\alpha : E \times B \to N$ is a morphism in $\mathbf{C}$,
- a morphism from $\langle E, B, \alpha\rangle$ to $\langle E', B', \alpha'\rangle$ is a pair $\langle f, F\rangle$ of morphisms in $\mathbf{C}$ such that $f : E \to E'$, $F : B' \to B$ and

$$\begin{tikzcd} E \times B' \arrow[r, "1 \times F"] \arrow[d, "f \times 1"'] \arrow[dr, phantom, "\geq"] & E \times B \arrow[d, "\alpha"] \\ E' \times B' \arrow[r, "\alpha'"'] & N, \end{tikzcd}$$

  where $\leq$ is the partial order induced pointwise on $\mathbf{C}(A, N)$ by the partial order $\leq$ on N, and
- composition is composition in $\mathbf{C}$ for each component, thus $\langle g, G\rangle\langle f, F\rangle = \langle gf, FG\rangle$,

[1]It is routine to generalise our construction by defining an order directly on the horn sets. For our purposes it suffices to consider concrete categories.

define a category, which we shall denote $\mathbf{M}_{\mathbb{N}}\mathbf{C}$.

> **Proof:** It is routine to verify that composition is well-defined, Let $\langle f, F\rangle : \langle E, B, \alpha\rangle \to \langle E', B', \alpha'\rangle$ and $\langle g, G\rangle : \langle E', B', \alpha'\rangle \to \langle E'', B'', \alpha''\rangle$ be morphisms in $\mathbf{M}_{\mathbb{N}}\mathbf{C}$. Then by definition $\alpha'(f \times 1) \leq \alpha(1 \times F)$ and $\alpha''(g \times 1) \leq \alpha'(1 \times G)$, we have $\alpha''(gf \times 1) \leq \alpha(1 \times FG)$, and $\langle gf, FG\rangle$ is a morphism in $\mathbf{M}_{\mathbb{N}}\mathbf{C}$ from $\langle E, B, \alpha\rangle$ to $\langle E'', B'', \alpha''\rangle$. Identities and associativity are inherited from $\mathbf{C}$.
> □

Since $\mathbf{C}$ is concrete, we interpret the ordering in the above diagram to mean that

$$\text{for each } e \in E \text{ and } b' \in B', \quad \alpha'\langle fe, b'\rangle \geq \alpha(e, Fb'\rangle.$$

Our construction is a variant on that of Chu [Bar79], in which the above diagram commutes. Chu's construction is applied to any category with a symmetric monoidal closed structure, and a distinguished object N. We can extend our construction to a symmetric monoidal category $\mathbf{C}$ [DP91] rather than a category with finite products, and this enables us (see Section 6) to capture the notion of morphism between elementary Petri nets studied in [NRT90], and also to consider other mathematically interesting categories [HDP90].

Remark 4.2 *Verity* [Ver91] *has proved that* $\mathbf{M}_{\mathbb{N}}\mathbf{C}$ *is enriched over* $\mathbf{C}$.

Remark 4.3 *It is evident that a categoy is also defined by the data:*

- *objects are the objects of* $\mathbf{M}_{\mathbb{N}}\mathbf{C}$,
- *a morphism from* $\langle E, B, \alpha\rangle$ *to* $\langle E', B', \alpha'\rangle$ *is a pair* $\langle f, F'\rangle$ *of morphisms in* $\mathbf{C}$ *such that* $f : E \to E',\ F : B' \to B$ *and*

$$\begin{tikzcd}
E \times B' \arrow[r, "1 \times F"] \arrow[d, "f \times 1"'] \arrow[dr, phantom, "\leq"] & E \times B \arrow[d, "\alpha"] \\
E' \times B' \arrow[r, "\alpha'"'] & N,
\end{tikzcd}$$

thus for each $e \in E$ and $b' \in B'$, we have $\alpha'\langle fe, b'\rangle \leq \alpha(e, Fb'\rangle$,

- *and composition given by composition in* $\mathbf{C}$ *for each component.*

We shall denote this category $\mathbf{M}_{\mathbb{N}}\mathbf{C}^*$.

Proposition 4.4 $\mathbf{M}_{\mathbb{N}}\mathbf{C}$ *is isomorphic to* $\mathbf{M}_{\mathbb{N}}\mathbf{C}^*$.

> **Proof:** There is an evident functor $\iota : \mathbf{M}_{\mathbb{N}}\mathbf{C} \to \mathbf{M}_{\mathbb{N}}\mathbf{C}^*$ taking the object $\langle E, B, \alpha\rangle$ to the object $\langle E, B, \alpha^{op}\rangle$[2] and taking the morphism $\langle f, F\rangle$ to $\langle F, f\rangle$.
> Thus
>
> $$\begin{tikzcd} E \times B' \arrow[r, "1 \times F"] \arrow[d, "f \times 1"'] \arrow[dr, phantom, "\geq"] & E \times B \arrow[d, "\alpha"] \\ E' \times B' \arrow[r, "\alpha'"'] & N \end{tikzcd} \quad \overset{\iota}{\longmapsto} \quad \begin{tikzcd} B' \times E \arrow[r, "1 \times f"] \arrow[d, "F \times 1"'] \arrow[dr, phantom, "\leq"] & B' \times E' \arrow[d, "(\alpha')^{op}"] \\ B \times E \arrow[r, "\alpha^{op}"'] & N. \end{tikzcd}$$
>
> It is clear that ι is an isomorphism. □

5 $\mathbf{M}_{\mathbb{N}}\mathbf{C}$ as a Model of Linear Logic

If the category $\mathbf{C}$ and the object N have appropriate structure, then we can define interesting structure on $\mathbf{M}_{\mathbb{N}}\mathbf{C}$. If $\mathbf{C}$ is Cartesian closed and has finite coproducts and $\langle N, \leq\rangle$ is a closed ordered monoid (that is, a partial order with a monotonic symmetric monoidal closed structure), then $\mathbf{M}_{\mathbb{N}}\mathbf{C}$ has finite products, finite coproducts and a symmetric monoidal closed structure. Note that these conditions are sufficient rather than necessary.

Part of the appeal of the category of elementary Petri nets **NSet** based on **GC** [BG90] lies in the fact that **NSet** is a sound model of linear logic in the sense of [DP89]. Given an interpretation of atomic formula as objects of **NSet**, we interpret the linear connective $\wedge$ by product, $\oplus$ by coproduct, $\otimes$ by the symmetric monoidal structure $\otimes$, linear implication $\multimap$ by the internal

[2]where $\alpha^{op}\langle b, e\rangle = \alpha\langle e, b\rangle$ for any $e \in E$ and $b \in B$

hom, *par* by a second symmetric monoidal structure $\Box$ and linear negation by the functor $(- \multimap \bot)$, where $\bot$ is the unit of $\Box$. Then whenever $\Gamma \vdash A$ in the fragment of linear logic comprising the structural rules, the rules for $\wedge, \oplus, \otimes, \multimap$ *par* and ! together with rules for intuitionistic negation, there is a morphism in **NSet** from the interpretation of Γ to the interpretation of A.

The category of Petri nets which we introduce in this paper is constructed from $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$, where **Set** is the category of sets and functions, and $\langle \mathbb{N}, \sqsubseteq \rangle$ is the set of natural numbers ordered by $\ldots 2 \sqsubseteq 1 \sqsubseteq 0$. Since **Set** is Cartesian closed and $\langle \mathbb{N}, \sqsubseteq \rangle$ is a closed ordered monoid (truncated subtraction being right adjoint to addition), $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$ has all finite products and coproducts, and is symmetric monoidal closed, affording a sound interpretation of intuitionistic linear logic.

6 General Nets and Simulation Morphisms

We now give a construction first sketched in [BG90]. We can regard a Petri net as an object $\langle \langle E, B, \mathit{pre} \rangle, \langle E', B', \mathit{post} \rangle \rangle$ of $\mathbf{M}_{\mathbb{N}}\mathbf{Set} \times \mathbf{M}_{\mathbb{N}}\mathbf{Set}$ for which $E = E'$ and $B = B'$. We can also regard it as an object of $\mathbf{M}_{\mathbb{N}}\mathbf{Set} \times \mathbf{M}_{\mathbb{N}}\mathbf{Set}^*$, $\mathbf{M}v\mathbf{Set}^* \times \mathbf{M}_{\mathbb{N}}\mathbf{Set}$ or $\mathbf{M}_{\mathbb{N}}\mathbf{Set}^* \times \mathbf{M}_{\mathbb{N}}\mathbf{Set}^*$. Thus each of these four categories gives rise to a category with object set **Petri**. Accordingly, we have four related notions of morphism between Petri nets. In Section 7 we give an elegant characterisation of any of these categories as a limit in **Cat**, the category of small categories and functors. This implies that all relevant structure of $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$ lifts to each of our categories of nets.

In our earlier work we indicated how the morphisms in **NSet**, a category with object set the elementary Petri nets, expressed refinement or simulation. We also varied our morphisms, obtaining the categories **NSet**, $\mathbf{NSet}^{*}$[3], $\mathbf{NSet}^{\leq}_{\geq}$ and $\mathbf{NSet}^{\geq}_{\leq}$. However, at that time it was unclear which notion of morphism was most appropriate.

In this paper we make precise the sense in which morphisms correspond to simulations, as Propositions 6.4 and 6.10 and Theorem 6.6 will demonstrate. A significant consequence of these results is that their proofs dictate the choices of the containments which we were previously unable to decide. Thus we shall work with the category with object set **Petri** which is a sub-

[3]called $\mathbf{NSet}^{co}$ in [Bro90]

category of $\mathbf{M}_{\mathbb{N}}\mathbf{Set} \times \mathbf{M}_{\mathbb{N}}\mathbf{Set}^*$, identifying the net $\langle E, B, pre, post\rangle$ with the object $\langle\langle E, B, pre\rangle, \langle B, E, post^{op}\rangle\rangle$. Note that we write $\leq$ for the usual ordering on $\mathbb{N}$.

Lemma 6.1 The following data:

- objects are general Petri nets, that is, elements of **Petri**,
- a morphism from $\langle E, B, pre, post\rangle$ to $\langle E', B', pre', post'\rangle$ is a pair of functions $\langle f, F\rangle$ with $f : E \to E'$ and $F : B' \to B$ such that

$$\begin{tikzcd}
E \times B' \arrow[r, "1 \times F"] \arrow[d, "f \times 1"'] \arrow[dr, phantom, "\leq"] & E \times B \arrow[d, "pre"] \\
E' \times B' \arrow[r, "pre'"'] & \mathbf{N}
\end{tikzcd}
\quad \text{and} \quad
\begin{tikzcd}
E \times B' \arrow[r, "1 \times F"] \arrow[d, "f \times 1"'] \arrow[dr, phantom, "\geq"] & E' \times B' \arrow[d, "post"] \\
E' \times B' \arrow[r, "post'"'] & \mathbf{N},
\end{tikzcd}$$

  that is, for each $e \in E$ and each $b' \in B'$, we have

$$pre\langle e, Fb'\rangle \geq pre'\langle fe, b'\rangle \text{ and } post\langle e, Fb'\rangle \leq post'\langle fe, b'\rangle,$$

- and composition is function composition in each component

define a category, which we shall denote **GNet**.

This result is a significant extension of that presented in [Bro90, BG90, BG], since it allows us to treat general nets rather than the subclass of elementary nets.

In addition, we can exploit the generality of our framework by replacing **Set** by **PSet**, the category of sets and partial functions, obtaining a category **PNet** with object set **Petri** which is a full subcategory of $\mathbf{M}_{\mathbb{N}}\mathbf{PSet} \times \mathbf{M}_{\mathbb{N}}\mathbf{PSet}^*$. As the proof of Theorem 7.3 is independent of the base category **C**, it is readily seen that **PNet** is symmetric monoidal closed and has finite products and coproducts. The subcategory of **PNet** in which our inequalities are replaced by equality, $\mathbf{PNet}^=$, has been studied in [NW91], and some relevant comparisons are made in [Bro90]. The full subcategory of $\mathbf{PNet}^=$ with objects the elementary Petri nets is that studied in [NRT90]. Thus our framework, starting from linear logic and an abstract approach, encompasses several existing models.

We now prove the results which determined our choice of morphism in **GNet**. We prove them for events and sequences of events: their extension to multisets of events and sequences of multisets of events is straightforward. We first make some definitions which enable us to discuss simulation in a precise way.

Definition 6.2 *Let $\langle N, M\rangle$ and $\langle N', M'\rangle$ be marked Petri nets, and let F be a function from B' to B. We say that the pair of markings $\langle M, M'\rangle$ is F-*`ok` *if $MF \leq M'$, that is, if we have*

$$\begin{tikzcd} B' \arrow[r, "F"] \arrow[d, "M'"'] \arrow[dr, phantom, "\geq"] & B \arrow[dl, "M"] \\ \mathbf{N} & \end{tikzcd} \qquad \textit{in } \mathbf{Set}$$

Definition 6.3 *Let $\langle f, F\rangle : N \to N'$ be a morphism in* **GNet**, *and for $i \in \{0, \dots n+1\}$ let $\langle M_i, M_i'\rangle$ be F-*`ok`. *Suppose that $M_0 \overset{A_0}{\rightsquigarrow}_1 M_1 \overset{A_1}{\rightsquigarrow}_1 M_2 \dots \overset{A_n}{\rightsquigarrow}_1 M_{n+1}$ in* N.
The direct simulation *of this evolution in* N$'$ *is $M_0' \overset{fA_0}{\rightsquigarrow}_1 M_1' \overset{fA_0}{\rightsquigarrow}_1 M_2' \dots \overset{fA_n}{\rightsquigarrow}_1 M_{n+1}'$.*
A simulation *of the above evolution in* N$'$ *is $M_0' \overset{s_0}{\rightsquigarrow}_1 M_1' \overset{s_1}{\rightsquigarrow}_1 M_2' \dots \overset{s_n}{\rightsquigarrow}_1 M_{n+1}'$, where for $i \in \{0, \dots, n\}$, s_i is a finite sequence of transitions t_{i_j} such that $\sum_j t_{i_j} = fA_i$.*

A weak simulation *of the above evolution in* N$'$ *is $M_0' \overset{v_0}{\rightsquigarrow}_1 M_1'' \overset{v_1}{\rightsquigarrow}_1 M_2'' \dots \overset{v_n}{\rightsquigarrow}_1 M_{n+1}''$, where for $i \in \{0, \dots n+1\}$, $\langle M_i, M_i''\rangle$ is F-*`ok` *and v_i is a finite sequence of transitions w_{i_j} such that $f^{-1}\sum_j w_{i_j} = fA_i$.*

Proposition 6.4 Let $\langle \mathrm{N}, M\rangle$ and $\langle \mathrm{N}', M'\rangle$ be marked Petri nets and let $\langle f, F\rangle$ be a morphism from N to N$'$ in **GNet** such that $\langle M, M'\rangle$ is F-`ok`. Then for all $e \in E$, $M \downarrow^{\mathrm{N}} e$ implies that $M' \downarrow^{\mathrm{N}'} fe$.

Proof: Suppose that $M \downarrow^{\mathrm{N}} e$, that is $\forall b \in B.\ (M(b) \geq pre\langle e, b\rangle)$.
In particular, $\forall b' \in B'$ we have $M(Fb') \geq pre\langle e, Fb'\rangle$.
However, by the definition of $\langle f, F\rangle$,

$$\forall e \in E, b' \in B' \text{ we have } pre\langle e, Fb'\rangle \geq pre'\langle fe, b'\rangle,$$

and therefore $\forall b' \in B'$ we have $M(Fb') \geq pre\langle e, Fb'\rangle \geq pre'\langle fe, b'\rangle$.
Now, by hypothesis, $\langle M, M'\rangle$ is F-`ok`, that is, $\forall b' \in B'.\ (M'(b') \geq M(Fb'))$, hence $\forall b' \in B'$ we have $M'(b') \geq M(Fb') \geq pre\langle e, Fb'\rangle \geq pre'\langle fe, b'\rangle$, which implies that $M' \downarrow^{\mathrm{N}'} fe$. □

If a pair of markings $\langle M, M'\rangle$ is F-`ok`, then the net N′ with marking M' can simulate any one-step evolution of the net N with marking M, in the sense that whenever $\langle \mathrm{N}, M\rangle$ enables an event e, $\langle \mathrm{N}', M'\rangle$ enables the event fe. We now show that this holds for any sequence of events in N, so that N′ can simulate any evolution of N.

Definition 6.5 *Let $\langle \mathrm{N}, M_0\rangle$ be a marked Petri net and let $e_0, e_1, \ldots, e_n \in E$. We say that $\langle \mathrm{N}, M_0\rangle$* enables *the sequence $e_0, e_1, \ldots, e_n$, written $M_0 \downarrow^{\mathrm{N}} e_0, e_1, \ldots, e_n$, if there is a sequence $M_0, M_1, \ldots, M_{n+1}$ of markings of* N *such that:*

$$\forall i \in \{0, \ldots, n\}.\ (M_i \downarrow^{\mathrm{N}} e_i \text{ and } M_i \overset{e_i}{\rightsquigarrow}_1 M_{i+1}).$$

The following important result shows that F-`ok`-ness is preserved under evolution. A consequence of this result is that if $\langle \mathrm{N}, M\rangle$ enables a sequence of events $e_0, e_1, \ldots, e_n$ then $\langle \mathrm{N}', M'\rangle$ enables the sequence $fe_0, fe_1, \ldots, fe_n$. Evidently for any marking M of N we can construct a marking M' of N′ such that $\langle M, M'\rangle$ is F-`ok`, and thus N′ can simulate any behaviour of N.

Theorem 6.6 Let $\langle \mathrm{N}, M_0\rangle$ and $\langle \mathrm{N}', M_0'\rangle$ be marked nets and let $\langle f, F\rangle$ be a morphism from N to N′ in **GNet**. If $\langle M_0, M_0'\rangle$ is F-`ok` and $M_0 \overset{e}{\rightsquigarrow}_1 M_1$, then $M_0' \overset{fe}{\rightsquigarrow}_1 M_1'$ and $\langle M_1, M_1'\rangle$ is F-`ok`.

Proof: Suppose that $\langle M_0, M_0'\rangle$ is F-`ok` and $M_0 \overset{e}{\rightsquigarrow}_1 M_1$. Then $M_0 \downarrow^{\mathrm{N}} e$ and therefore, by Proposition 6.4, $M_0' \downarrow^{\mathrm{N}'} fe$ and $M_0' \overset{fe}{\rightsquigarrow}_1 M_1'$.
Now $M_1 = M_0 - pre(e) + post(e)$ and $M_1' = M_0' - pre'(fe) + post'(fe)$.
That is:

$\forall b \in B.\ (M_1(b) = (M_0(b) - \mathit{pre}\langle e, b\rangle)^4 + \mathit{post}\langle e, b\rangle)$ and
$\forall b' \in B'.\ (M'_1(b') = (M'_0(b') - \mathit{pre}'\langle fe, b'\rangle) + \mathit{post}'\langle fe, b'\rangle)$.

Now for each $b' \in B'$, we have

$$\begin{aligned}
M'_1(b') &= (M'_0(b') - \mathit{pre}'\langle fe, b'\rangle) + \mathit{post}'\langle fe, b'\rangle \\
&\geq (M_0(Fb') - \mathit{pre}'\langle fe, b'\rangle) + \mathit{post}'\langle fe, b'\rangle \quad \text{as } \langle M_0, M'_0\rangle\ \mathit{is} F{-}\texttt{ok} \\
&\geq (M_0(Fb') - \mathit{pre}'\langle e, Fb'\rangle) + \mathit{post}'\langle fe, b'\rangle \quad \text{by definition of } \langle f, F\rangle \\
&\geq (M_0(Fb') - \mathit{pre}'\langle e, Fb'\rangle) + \mathit{post}\langle e, Fb'\rangle \quad \text{by definition of } \langle f, F\rangle \\
&= M_1(Fb').
\end{aligned}$$

Thus $\forall b' \in B'$ we have $M'_1(b') \geq M_1(Fb')$, and so $\langle M_1, M'_1\rangle$ is F-`ok`. □

Corollary 6.7 Let $\langle \mathrm{N}, M_0\rangle$ and $\langle \mathrm{N}', M'_0\rangle$ be marked Petri nets and let $\langle f, F\rangle$ be a morphism from N to N′ in **GNet**. If $\langle M_0, M'_0\rangle$ is F-`ok`, then whenever $M_0 \downarrow^{\mathrm{N}} e_0, e_1, \ldots, e_n$, we have $M'_0 \downarrow^{\mathrm{N}'} fe_0, fe_1, \ldots, fe_n$.

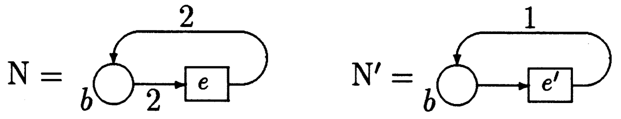
Proof: By induction on the length of the sequence. For the inductive step, observe that if $M_i \downarrow^{\mathrm{N}} e_i$ and $M_i \overset{e_i}{\rightsquigarrow}_1 M_{i+1}$ and $\langle M_i, M'_i\rangle$ is F-`ok` then $M'_i \downarrow^{\mathrm{N}'} fe_i$ (by Proposition 6.4), while $M'_i \overset{fe_i}{\rightsquigarrow}_1 M'_{i+1}$ and $\langle M_{i+1}, M'_{i+1}\rangle$ is F-`ok` (by Theorem 6.6). □

Thus if $\langle \mathrm{N}, M\rangle$ enables a sequence of events $e_0, e_1, \ldots, e_n$ then $\langle \mathrm{N}', M'\rangle$ enables the sequence $fe_0, fe_1, \ldots, fe_n$. Evidently for any marking M of N we can construct a marking M' of N′ such that $\langle M, M'\rangle$ is F-`ok`, and thus N′ can simulate any behaviour of N. Recall that a weak simulation of an evolution may include events which are not in the image of f (that is, events which do not correspond directly to any event in the simulated net) but while it evolves under such events, F-`ok`-ness is preserved and so the simulating net N′ never loses the capacity to proceed with the direct simulation of the evolution of N. Note that a direct simulation is a simulation, and that a simulation is a weak simulation. The concept of weak simulation allows us to consider simulations in which the simulated net N may idle (or stutter) while the simulating net proceeds with events which do not correspond directly to events of N. In Section 6.1 we give an example to illustrate these concepts. The following proposition shows that we can extend the results above from direct simulations to weak simulations. The extension of Proposition 6.8 from events to transitions is straightforward.

Proposition 6.8 Let $\langle f, F\rangle : \mathrm{N} \to \mathrm{N}'$ be a morphism in **GNet**. Let the pair $\langle M_0, M_0'\rangle$ be F-`ok`. If $M_0 \downarrow^N e_0, e_1, \ldots, e_n$ then M_0' enables any weak simulation of $e_0, e_1, \ldots, e_n$.

> **Proof:** We show that whenever $M_0 \overset{e}{\rightsquigarrow}_1 M_1$, there is a weak simulation $M_0' \overset{*}{\rightsquigarrow}_1 M_a' \overset{fe}{\rightsquigarrow}_1 M_b' \overset{*}{\rightsquigarrow}_1 M_c'$, where $*$ stands for any transition of N′ which is disjoint from $f(E)$, and $\langle M_1, M_c'\rangle$ is F-`ok`.
> If $M_0' \overset{*}{\rightsquigarrow}_1 M_a'$ then by definition of weak simulation, $\langle M_0, M_a'\rangle$ is F-`ok`. By Proposition 6.4, M_a' enables fe.
> If $M_a' \overset{fe}{\rightsquigarrow}_1 M_b'$ then, by Theorem 6.6, the pair $\langle M_1, M_b'\rangle$ is F-`ok`. By definition of weak simulation, if $M_b' \overset{*}{\rightsquigarrow}_1 M_c'$ then $\langle M_1, M_c'\rangle$ is F-`ok`.
> We can repeat this argument for each of the transitions of the evolution $e_0, e_1, \ldots, e_n$, and the result follows. $\square$

Theorem 6.6 showed that whenever we have a morphism from N to N′ in **GNet**, N′ can simulate any evolution of N. Ideally, we would like to have a converse to this result. That is, a result which states that if N′ can simulate any evolution of N then there is a morphism from N to N′ in **GNet**, since this would show that our morphisms exactly capture the independent notion of simulation. Unfortunately, as the following example shows, we cannot quite achieve this.



N′ can simulate any evolution of N but there is no morphism from N to N′ because $post(e, Fb) = 2 > 1 = post'(fe, b)$ It transpires that the problem is that the first net has a condition, b, which is both a pre-condition and a post-condition of some event. If we restrict ourselves to the case where the first net does not have this property then we do indeed have a converse to Theorem 6.6.

Definition 6.9 *A net is* loop free *if for all events e, $pre(e) \cap post(e) = \emptyset$.*

Proposition 6.10 Let N be a loop free net, let N$'$ be any net, let $f : E \to E'$ and $F : B' \to B$, and suppose that for all F-`ok` pairs of markings $\langle M, M' \rangle$,

- $M \downarrow e$ implies that $M' \downarrow f(e)$, and
- if $M \overset{e}{\rightsquigarrow} M_1$ and $M' \overset{f(e)}{\rightsquigarrow} M_1'$ then $\langle M_1, M_1' \rangle$ is F-`ok`

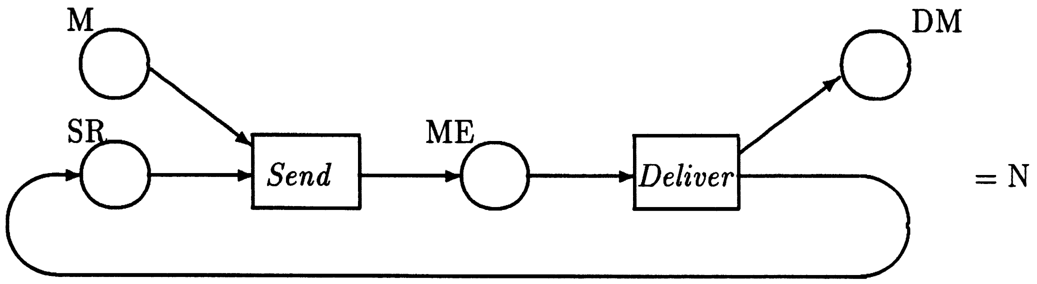
then $\langle f, F \rangle$ is a morphism from N to N$'$ in **GNet**.

> **Proof:** Let $M = \mathit{pre}(e)$ and $M' = MF$, so $\langle M, M' \rangle$ is clearly F-`ok`. Then $M \downarrow^{\mathrm{N}} e$ whence, by assumption, $M' \downarrow^{\mathrm{N}'} f(e)$ and so for all $b' \in B'$, $\mathit{pre}'(fe, b') \leq M'(b') = M(Fb') = \mathit{pre}(e, Fb')$.
>
> Now $M \overset{e}{\rightsquigarrow} \mathit{post}(e) = M_1$ and $M' = MF \overset{f(e)}{\rightsquigarrow} (MF - \mathit{pre}'(fe)) + \mathit{post}'(fe) = M_1'$ and by assumption $\langle M_1, M_1' \rangle$ is F-`ok`. That is,
>
> $M_1(Fb') = \mathit{post}(e, Fb') \leq (\mathit{pre}(e, Fb') - \mathit{pre}'(fe, b')) + \mathit{post}'(fe, b')$.
>
> If $\mathit{post}(e, Fb') = 0$ then $\mathit{post}(e, Fb') \leq \mathit{post}'(fe, b')$ and we are done. Otherwise, $\mathit{pre}(e, Fb') = 0$ by the assumption that N is loop free. However, we have shown that $\mathit{pre}(e, Fb') \geq \mathit{pre}'(fe, b')$ and so $\mathit{pre}'(fe, b') = 0$ as well whence,
>
> $$\mathit{post}(e,Fb') \leq (\mathit{pre}(e,Fb') - \mathit{pre}'(fe,b')) + \mathit{post}'(fe,b') = (0-0) + \mathit{post}'(fe,b') = \mathit{post}'(fe,b').$$
>
> Thus $\langle f, F \rangle$ is a morphism from N to N$'$ in **GNet**. $\square$

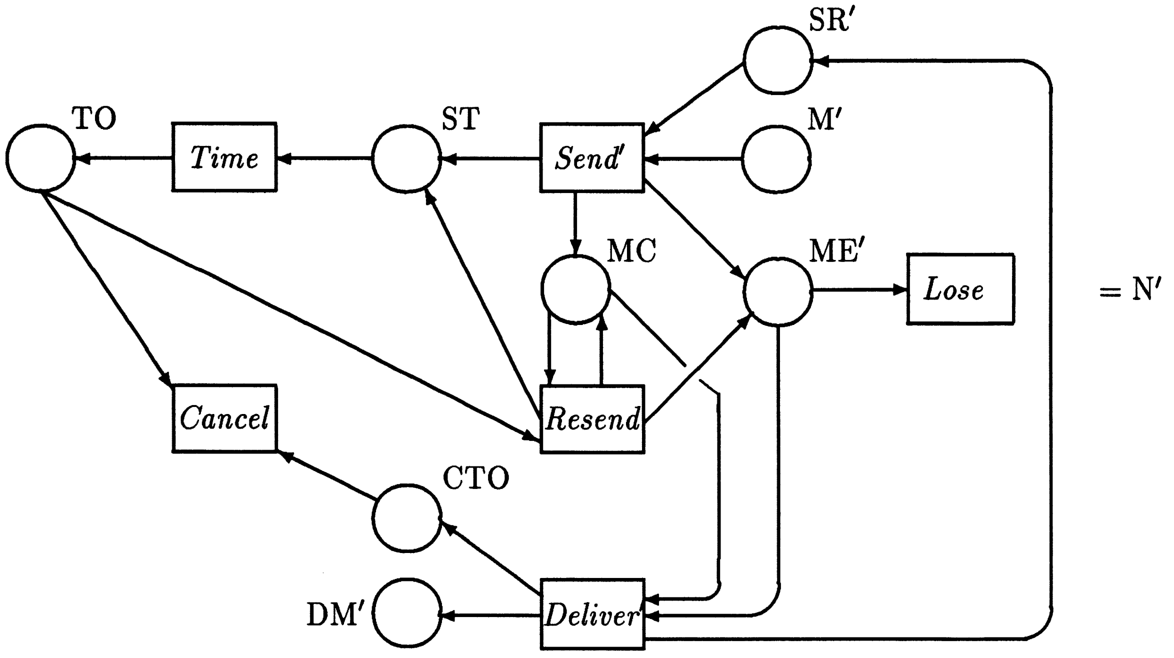
The results of this section are important because they show that, not only do the marphisms of **GNet** have a meaningful computational interpretation, but also that [between loop-free nets] all simulations are captured by our morphisms. In other approaches [Win84, MM88b, NRT90], many simulations are not expressible as morphisms.

6.1 A Practical Example of a Simulation

We shall now exhibit a morphism from an unintelligent message handler to a more sophisticated message handler which can correct for the loss of a message in transit. This suffices to show that the message handler with error-correction can exhibit the simple behaviour of successively sending and delivering messages.



The simple message handler, with net N, takes a message (M) and a flag that the sender is ready (SR), and dispatches the message to the ether via the event *Send*. The event *Deliver* accepts a message from the ether (ME) and delivers it (DM) to its destination, meanwhile setting the flag to indicate that the sender is once more ready to send a message. This net does not model any of the problems associated with a real message handler, a significant problem being the loss of messages from the ether.



The error-correcting message handler resends a message which has been lost, and its net, N′, is more complex. The event *Send′* takes a message (M′) and a flag that the sender is ready (SR′), creates a copy of the message (MC) and dispatches the message to the ether (ME$'$), meanwhile starting a timer (ST). The handler may *Lose* the message. Alternatively, the event *Deliver$'$* may take a message from the ether, delete its copy, deliver it (DM$'$), set a flag (CTO) cancelling the timeout signal and set the flag SR$'$ indicating that the sender is ready. The event *Time* is started by ST, and after a certain amount of time produces a timeout signal (TO). If a timeout signal is produced after the message has been delivered, CTO is set and the event *Cancel* forgets the timeout, resetting the flag CTO. If CTO is not marked when *Time* times out, then *Cancel* cannot occur but *Resend* becomes enabled. On receiving the timeout, *Resend* takes a message copy, recopies it and dispatches it to the ether, meanwhile restarting the timer.

Thus the message handler N$'$ can take a message and deliver it. If the message has not been delivered after a certain amount of time, it will resend the message. This enables it to correct the error of a message lost in the ether. It may also result in superfluous copies of a message being sent, since a timeout may occur when no message has been lost. To handle this error we would need a still more refined system, such as the alternating bit protocol [Mil89].

There is a morphism $\langle f, F\rangle$ in **GNet** from N to N$'$ given as follows:

$$\begin{array}{lll} f(\mathit{Send}) = \mathit{Send}' & & f(\mathit{Deliver}) = \mathit{Deliver}' \\ F(\mathrm{SR}') = \mathrm{SR} & & F(\mathrm{M}') = F(\mathrm{TO}) = \mathrm{M} \\ F(\mathit{ME}') = F(\mathrm{MC}) = F(\mathrm{ST}) = \mathrm{ME} & \text{and} & F(\mathrm{CTO}) = F(\mathrm{DM}') = \mathrm{DM}. \end{array}$$

It is straightforward to check that the functions f and F defined above satisfy the required conditions, that is, for each event e of N and each condition b' of N$'$, we have

$$\mathit{pre}\langle e, Fb'\rangle \geq \mathit{pre}'\langle fe, b'\rangle \quad\text{and}\quad \mathit{post}(e, Fb') \leq \mathit{post}'\langle fe, b'\rangle.$$

The inequality is strict in two cases.

Theorem 6.6 indicates that for any pair $\langle M, M'\rangle$ of F-`ok` markings, N$'$ with marking M' can simulate any behaviour of N with marking M. For example, N can evolve from marking 2M + SR as follows:

$$2\mathrm{M} + \mathrm{SR} \overset{\mathit{Send}}{\rightsquigarrow_1} \mathrm{M} + \mathrm{ME} \overset{\mathit{Deliver}}{\rightsquigarrow_1} \mathrm{M} + \mathrm{DM} + \mathrm{SR} \overset{\mathit{Send}}{\rightsquigarrow_1} \mathrm{ME} + \mathrm{DM} + \mathrm{SR} \overset{\mathit{Deliver}}{\rightsquigarrow_1} 2\mathrm{DM} + \mathrm{SR}.$$

The direct simulation of this evolution in N$'$ is

$$\begin{aligned}2\mathrm{M}' + \mathrm{SR}' &\overset{\mathit{Send}'}{\rightsquigarrow_1} \mathrm{M}' + \mathrm{ME}' + \mathit{ST} + \mathit{MC} \overset{\mathit{Deliver}'}{\rightsquigarrow_1} \mathrm{M}' + \mathrm{DM}' + \mathrm{SR}' + \mathrm{ST} + \mathrm{CTO}\\ &\overset{\mathit{Send}'}{\rightsquigarrow_1} \mathrm{ME}' + \mathrm{DM}' + \mathrm{MC} + 2\mathrm{ST} + \mathrm{CTO} \overset{\mathit{Deliver}'}{\rightsquigarrow_1} 2\mathrm{DM}' + 2\mathrm{ST} + 2\mathrm{CTO}.\end{aligned}$$

It is easy to check that at each stage in this evolution, the corresponding pair of markings $\langle M, M' \rangle$ is F-`ok`. The fact that the final marking of N′ includes the marking $2\mathrm{ST} + 2\mathrm{CO}$ conflicts somewhat with our intuition. Because the events *Time* and *Cancel* are not in the image of f, they do not occur in the direct simulation. The evolution

$$\begin{aligned}2\mathrm{M}' + \mathrm{SR}' &\overset{\mathit{Send}'}{\rightsquigarrow_1} \mathrm{M}' + \mathrm{ME}' + \mathit{ST} + \mathit{MC} \overset{\mathit{Deliver}'}{\rightsquigarrow_1} \mathrm{M}' + \mathrm{DM}' + \mathrm{SR}' + \mathrm{ST} + \mathrm{CTO}\\ &\overset{\mathit{Time}}{\rightsquigarrow_1} \mathrm{M}' + \mathrm{DM}' + \mathrm{SR}' + \mathrm{TO} + \mathrm{CTO} \overset{\mathit{Cancel}}{\rightsquigarrow_1} \mathrm{M}' + \mathrm{DM} + \mathrm{SR}'\\ &\overset{\mathit{Send}'}{\rightsquigarrow_1} \mathrm{ME}' + \mathrm{DM}' + \mathrm{MC} + \mathrm{ST} \overset{\mathit{Deliver}'}{\rightsquigarrow_1} 2\mathrm{DM}' + \mathrm{ST} + \mathrm{CTO}\\ &\overset{\mathit{Time}}{\rightsquigarrow_1} 2\mathrm{DM}' + \mathrm{SR}' + \mathrm{TO} + \mathrm{COT} \overset{\mathit{Cancel}}{\rightsquigarrow_1} 2\mathrm{DM}' + \mathrm{SR}'\end{aligned}$$

of N′ *weakly* simulates the evolution

$$2\mathrm{M} + \mathrm{SR} \overset{\mathit{Send}}{\rightsquigarrow_1} \mathrm{M} + \mathrm{ME} \overset{\mathit{Deliver}}{\rightsquigarrow_1} \mathrm{M} + \mathrm{DM} + \mathrm{SR} \overset{\mathit{Send}}{\rightsquigarrow_1} \mathrm{ME} + \mathrm{DM} + \mathrm{SR} \overset{\mathit{Deliver}}{\rightsquigarrow_1} 2\mathrm{DM} + \mathrm{SR}.$$

Various other evolutions of N′ also evolve to marking $2\mathrm{DM}' + \mathrm{SR}'$ while preserving F-`ok`-ness throughout.

The fact that there is a morphism in **GNet** from N to N′ proves that N′ correctly implements the behaviour specified by N. This motivates our choice of morphism in **GNet**.

6.2 CCS Simulation

We have described the morphisms in **GNet** as simulations on the grounds that a morphism between two nets N and N′ verifies that N′ can simulate any behaviour of N. In fact, we can also understand our morphisms as simulations analogous to those of labelled transition systems such as CCS [Mil89]. We recall the definition of a strong bisimulation in CCS.

Definition 6.11 *Let $\langle \mathcal{P}, \mathcal{A} \rangle$ be a labelled transition system. A binary relation $S \subseteq \mathcal{P} \times \mathcal{P}$ is a* strong bisimulation *if $(P, Q) \in S$ implies that, for all $\alpha \in \mathcal{A}$:*

- *whenever* $P \xrightarrow{\alpha} P'$ *then, for some* $Q', Q \xrightarrow{\alpha} Q'$ *and* $(P', Q') \in S$ *and*
- *whenever* $Q \xrightarrow{\alpha} Q'$ *then, for some* $P', P \xrightarrow{\alpha} P'$ *and* $(P', Q') \in S$.

Similarly, one can define a simulation as follows.

Definition 6.12 *Let* $\langle \mathcal{P}, \mathcal{A} \rangle$ *be a labelled transition system. A binary relation* $S \subseteq \mathcal{P} \times \mathcal{P}$ *is a* simulation *if* $(P, Q) \in S$ *implies that, for all* $\alpha \in \mathcal{A}$*:*

- *whenever* $P \xrightarrow{\alpha} P'$ *then, for some* $Q', Q \xrightarrow{\alpha} Q'$ *and* $(P', Q') \in S$.

In fact, we require a slight generalisation of this definition which extends simulation to a relation between states of different transition systems.

Definition 6.13 *Let* $\langle \mathcal{P}, \mathcal{A} \rangle$ *and* $\langle \mathcal{P}', \mathcal{A}' \rangle$ *be labelled transition systems and let* f *be a function from* $\mathcal{A}$ *to* $\mathcal{A}'$*. A binary relution* $S \subseteq \mathcal{P} \times \mathcal{P}'$ *is a* simulation relative to f *if* $(P, Q) \in S$ *implies that, for all* $\alpha \in \mathcal{A}$:

- *whenever* $P \xrightarrow{\alpha} P'$ *then, for some* $Q' \in \mathcal{P}', Q \xrightarrow{f\alpha} Q'$ *and* $(P', Q') \in S$.

Now, it is well known that a Petri Net N may be viewed as a transition system $\langle \mathit{Mark}(\mathrm{N}), E \rangle$ with transition relation given by $M_0 \xrightarrow{e} M_1$ if $M_0 \downarrow^N e$ and $M_1 = M_0 - \mathit{pre}(e) + \mathit{post}(e)$. Observe that F-`ok` is a relation between $\mathit{Mark}(\mathrm{N})$ and $\mathit{Mark}(\mathrm{N}')$.

Proposition 6.14 Let $\langle f, F \rangle$ be a morphism from a net N to a net N′ in **GNet**. Then F-`ok` is a simulation relative to f between $\langle \mathit{Mark}(\mathrm{N}), E \rangle$ and $\langle \mathit{Mark}(\mathrm{N}'), E' \rangle$.

> **Proof:** Follows immediately from Proposition 6.4 and Theorem 6.6. □

Moreover, we can generalise the definition of strong bisimulation in an analogous way:

Definition 6.15 *Let* $\langle \mathcal{P}, \mathcal{A} \rangle$ *and* $\langle \mathcal{P}', \mathcal{A}' \rangle$ *be labelled transition systems, let* f *be a function from* $\mathcal{A}$ *to* $\mathcal{A}'$ *and let* g *be a function from* $\mathcal{A}'$ *to* $\mathcal{A}$*. A binary relation* $S \subseteq \mathcal{P} \times \mathcal{P}'$ *is a* bisimulation relative to (f, g) *if* $(P, Q) \in S$ *implies that, for all* $\alpha \in \mathcal{A}, \alpha' \in \mathcal{A}'$*:*

- *whenever* $P \xrightarrow{\alpha} P'$ *then, for some* $Q' \in \mathcal{P}', Q \xrightarrow{f\alpha} Q'$ *and* $(P', Q') \in S$ and
- *whenever* $Q \xrightarrow{\alpha'} Q'$ *then, for some* $P' \in \mathcal{P}, Q \xrightarrow{g\alpha'} Q'$ *and* $(P', Q') \in S$.

Proposition 6.16 Let N and N$'$ be nets, let $\langle f, F\rangle$ be a morphism from N to N$'$ in **GNet** and let $\langle g, G\rangle$ be a morphism from N$'$ to N in **GNet**. Then the relation $\sim \subseteq \mathit{Mark}(\mathrm{N}) \times \mathit{Mark}(\mathrm{N}')$ given by:

$$M \sim M' \;\text{ if }\; \langle M, M'\rangle \text{ is } F\text{-}\mathtt{ok} \;\text{ and }\; \langle M', M\rangle \text{ is } G\text{-}\mathtt{ok}$$

is a bisimulation relative to (f, g).

7 Structure in GNet

The category $\mathbf{M_{\mathbb{N}}Set}$, and the isomorphic category $\mathbf{M_{\mathbb{N}}Set}^*$ have considerable categorical structure. In particular, they have finite products and coproducts and a symmetric monoidal closed structure [DP91]. An important issue is the extent to which this structure lifts to our category **GNet**. We now give an elegant characterisation of **GNet** as a limit in **Cat**, which leads to an easy proof that **GNet** has finite products, finite coproducts and a symmetric monoidal structure induced by those in $\mathbf{M_{\mathbb{N}}Set}$. The proof closely follows the proof that **NC** inherits the structure of **GC** (see [Bro90, BG90, BG]).

Lemma 7.1 **GNet** is the pullback in **Cat** of the forgetful functor $\mathcal{U}$: $\mathbf{M_{\mathbb{N}}Set}$ + $\mathbf{Set} \times \mathbf{Set}^{op}$ along itself (a kernel pair).

> **Proof:** First recall that $\mathbf{M_{\mathbb{N}}Set}$ is isomorphic to $\mathbf{M_{\mathbb{N}}Set}^*$, and pullbacks are only defined up to isomorphism. Thus the pullback in **Cat** of $\mathcal{U}$: $\mathbf{M_{\mathbb{N}}Set} \to \mathbf{Set} \times \mathbf{Set}^{op}$ along $\mathcal{U}^*$: $\mathbf{M_{\mathbb{N}}Set}^* \to \mathbf{Set} \times \mathbf{Set}^{op}$ is isomorphic to the pullback:

$$\begin{tikzcd}
\mathbf{X} \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \mathbf{M_{\mathbb{N}}Set} \arrow[d, "\mathcal{U}"] \\
\mathbf{M_{\mathbb{N}}Set} \arrow[r, "\mathcal{U}"'] & \mathbf{Set} \times \mathbf{Set}^{op}
\end{tikzcd}$$

where $\mathcal{U}(\langle E, B, \alpha\rangle) = \langle E, B\rangle$ and $\mathcal{U}(\langle f, F\rangle) = \langle f, F\rangle$. Since **Cat** is Cartesian, this pullback is the equaliser of the arrows $\mathcal{U}\pi_0$ and $\mathcal{U}\pi_1$ from $\mathbf{M_{\mathbb{N}}Set} \times \mathbf{M_{\mathbb{N}}Set}$ to $\mathbf{Set} \times \mathbf{Set}^{op}$. Hence

$$Ob(\mathbf{X}) = \{\langle A_0, A_1\rangle \in Ob(\mathbf{M_{\mathbb{N}}Set}) \times Ob(\mathbf{M_{\mathbb{N}}Set}) \mid \mathcal{U}A_0 = \mathcal{U}A_1\},$$

and similarly an arrow in $\mathbf{X}$ from $\langle A_0, A_1\rangle$ to $\langle A_0', A_1'\rangle$ is a pair $\langle\langle f, F\rangle, \langle g, G\rangle\rangle$ with $\langle f, F\rangle : A_0 \to A_0'$ in $\mathbf{M_{\mathbb{N}}Set}$ and $\langle g, G\rangle : A_1 \to A_1'$ in $\mathbf{M_{\mathbb{N}}Set}$ such that $\mathcal{U}\langle f, F\rangle = \mathcal{U}\langle g, G\rangle$, that is, such that $f = g$ and $F = G$.

Thus objects of $\mathbf{X}$ are of form $\langle\langle E, B, \alpha\rangle, \langle E, B, \alpha'\rangle\rangle$ and a morphism in $\mathbf{X}$ from $\langle\langle E, B, \alpha_0\rangle, \langle E, B, \alpha_1\rangle\rangle$ to $\langle\langle E', B', \alpha_0'\rangle, \langle E', B', \alpha_1'\rangle\rangle$ is a pair $\langle\langle f, F\rangle, \langle f, F\rangle\rangle$ such that $\langle f, F\rangle : \langle E, B, \alpha_0\rangle \to \langle E', B', \alpha_0'\rangle$ and $\langle f, F\rangle : \langle E, B, \alpha_1\rangle \to \langle E', B', \alpha_1'\rangle$ in $\mathbf{M_{\mathbb{N}}Set}$. Evidently, $\mathbf{X}$ is isomorphic to **GNet**. $\Box$

Lemma 7.2 $\mathcal{U}$ strictly preserves the product, coproduct and symmetric monoidal closed structure of $\mathbf{M_{\mathbb{N}}Set}$.

Proof: Straightforward (compare the corresponding proof in [Bro90]). $\Box$

Theorem 7.3 GNet has the products, coproducts and symmetric monoidal closed structure induced by those in $\mathbf{M_{\mathbb{N}}Set}$.

Proof: The category of small categories with assigned finite products, assigned finite coproducts and strict monoidal closed structure, with morphisms the functors strictly preserving this structure, is monadic over **Cat**. The evident forgetful functor from it into **Cat** creates all limits, including kernel pairs. Since the product, coproduct and symmetric monoidal closed structure of $\mathbf{M_{\mathbb{N}}Set}$ are strictly preserved by $\mathcal{U}$ (Lemma 7.2), and since **GNet** is the kernel pair of $\mathcal{U}$ (Lemma 7.1), **GNet** has the products, coproducts and symmetric monoidal closed structure induced by those in $\mathbf{M_{\mathbb{N}}Set}$. $\Box$

This result has two consequences. Firstly, we maintain the connection with linear logic, since **GNet** is a sound model of the fragment of linear logic comprising the structural rules and the rules for $\wedge, \oplus, \otimes$ and $\multimap$. Secondly, we have obtained several constructors on general Petri nets. If N and N' are nets, then their product $\mathrm{N} \times \mathrm{N}'$, their coproduct $\mathrm{N} + \mathrm{N}'$, their tensor product $\mathrm{N} \otimes \mathrm{N}'$ and the implication $\mathrm{N} \multimap \mathrm{N}'$ can all be interpreted as nets. To interpret the evolution of the compound net, we require an appropriate way of combining the markings of its component nets. In general, several possibilities exist, and decisions are hard to make using intuition alone. In Section 10 we resolve this issue using a category of marked nets.

7.1 The Product of two Nets

As we have seen, the product in **GNet** is induced by that in $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$. Thus the product in **GNet** of nets $\langle E_0, B_0, \mathit{pre}_0, \mathit{post}_0\rangle$ and $\langle E_1, B_1, \mathit{pre}_1, \mathit{post}_1\rangle$ is $\langle E_0 \times E_1, B_0 + B_1, \mathit{pre}, \mathit{post}\rangle$, where

$$\begin{aligned} \mathit{pre}\langle e_0, e_1\rangle &= \sum_{b_0 \in B_0} \mathit{pre}_0\langle e_0, b_0\rangle (b_0, 0) + \sum_{b_1 \in B_1} \mathit{pre}_1\langle e_1, b_1\rangle (b_1, 1), \quad \text{and} \\ \mathit{post}\langle e_0, e_1\rangle &= \sum_{b_0 \in B_0} \mathit{post}_0\langle e_0, b_0\rangle (b_0, 0) + \sum_{b_1 \in B_1} \mathit{post}_1\langle e_1, b_1\rangle (b_1, 1). \end{aligned}$$

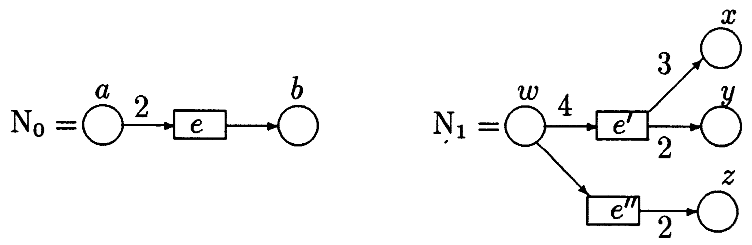
The precise details of the construction are given in Appendix B.

An event in the product net is the synchronisation of two events, one in each of the component nets: that is, a firing in the product net is the concurrent firing of an event in the first component and an event in the second. We relate the behaviour of the net $\mathrm{N}_0 \times \mathrm{N}_1$ to that of its component nets neatly via Proposition 7.4, which shows that the behaviour of a product net is the product of the behaviours of its component nets.

Proposition 7.4 A marking M of $\mathrm{N}_0 \times \mathrm{N}_1$ can evolve by a multiset of events A over E to a marking M' if and only if for $i = 0$ and $i = 1$, the marking $\mathit{in}_i^{op} M$ can evolve in N_i by the multiset of events $\pi_i A$ to the marking $\mathit{in}_i^{op} M'$.

Constructing products is essential for modelling parallel compositions. We give a simple example of the product of two nets. We give a more complex example of the use of the product in Section 8.

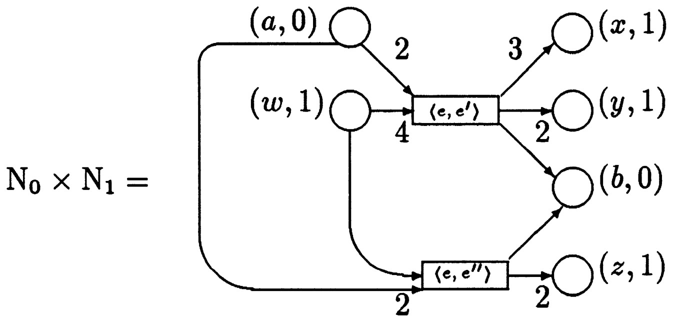
Example 7.5 Consider the two nets N_0 and N_1 given below:



The product net $N_0 \times N_1$ has event set $E = \{\langle e, e'\rangle, \langle e, e''\rangle\}$ and condition set $B = \{(a,0),(b,0),(w,1),(x,1),(y,1),(z,1)\}$. The pre- and post-condition relations of $N_0 \times N_1$ are given by:

$$\begin{aligned} &pre\langle e, e'\rangle = 2(a,0) + 4(w,1), && post\langle e, e'\rangle = (b,0) + 3(x,1) + 2(y,1),\\ &pre\langle e, e''\rangle = 2(a,0) + (w,1), \quad \text{and} && post\langle e, e''\rangle = (b,0) + 2(z,1). \end{aligned}$$

Thus the product net $N_0 \times N_1$ is given by:



Remark 7.6 *The product of two nets in* **GNet** *is their synchronous product* [Win87].

7.2 The coproduct of two nets

The coproduct in **GNet** is also induced by that in $\mathbf{M_{\mathbb{N}}Set}$. Thus the coproduct in **GNet** of nets $\langle E_0, B_0, pre_0, post_0\rangle$ and $\langle E_1, B_1, pre_1, post_1\rangle$ is $\langle E_0 + E_1, B_0 \times B_1, pre, post\rangle$, where

$$\begin{aligned} pre(e_0, 0) &= \sum_{b_0 \in B_0} \sum_{b_1 \in B_1} pre_0\langle e_0, b_0\rangle \langle b_0, b_1\rangle \text{ and}\\ post(e_0, 0) &= \sum_{b_0 \in B_0} \sum_{b_1 \in B_1} post_0\langle e_0, b_0\rangle \langle b_0, b_1\rangle, \end{aligned}$$

and similarly for $(e_1, 1)$. In general the coproduct is not an appealing construct. However, we make considerable use of the special case $N + \perp$, where

$\perp = \langle 1, 1, 0, o \rangle$. Thus $\perp$ is the net consisting of one event $*$ and one condition $*$, with zero pre- and post-condition relations: it is the unit of $\otimes$, the symmetric monoidal structure induced in **GNet** by that in $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$ (described in Appendix A). We shall call the event with empty pre- and post-conditions the *idling event*, denoted $*$ or $\langle *, * \rangle$ according to context. Now we have

$$\mathrm{N} + \perp = \langle E + \{*\}, B \times \{*\}, pre', post' \rangle, \text{ where } pre'(e_0, 0) = \sum_{b \in B} pre(e, b)\langle b, * \rangle \cong pre(e),$$

and so on. For brevity, we shall write $\mathrm{N}_\perp$ for $\mathrm{N} + \perp$ and identify $E + \{*\}$ with $E \cup \{*\}$, and $B \times \{*\}$ with B.

7.3 Expressing Synchrony and Asynchrony

We have seen that an event $\langle e_0, e_1 \rangle$ in the product net $\mathrm{N}_0 \times \mathrm{N}_1$ is enabled whenever the events e_0 and e_1 in the component nets are both enabled at the same time. Thus events of the product net correspond to *synchronisations* of events in the component nets. Our next example demonstrates how events may occur asynchronously in a product net.

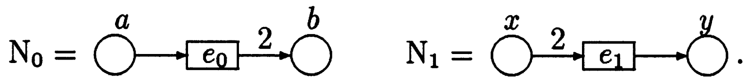
Definition 7.7
An event $\langle e_0, e_1 \rangle$ of the product net $\mathrm{N}_0 \times \mathrm{N}_1$ is asynchronous *if for either $i = 0$ or $i = 1$,*

$$pre\langle e_0, e_1 \rangle = in_i(pre(e_i)) \quad and \quad post\langle e_0, e_1 \rangle = in_i(post(e_i)),$$

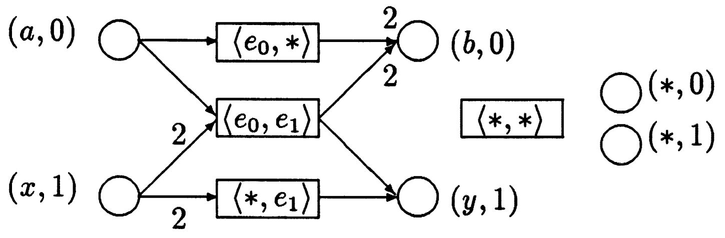
where for $i = 0, 1$ we extend the function in_i to multisets over B_i in the evident way.

Thus an asynchronous event is an event of a product net whose firing concerns only one of the component nets of the product. We shall now illustrate how asynchronous events arise when we apply our categorical constructions to nets.

Example 7.8 Let N_0 and N_1 be the nets:



The product net $(\mathrm{N}_0 + \bot) \times (\mathrm{N}_1 + \bot)$ is given by:



Observe that $pre\langle e_0,*\rangle = in_0(pre(e)$ *and* $post\langle e_0,*\rangle) = in_0(post(e))$. Thus $\langle e_0,*\rangle$, and similarly $\langle *,e_1\rangle$, are asynchronous events in the product net $(\mathrm{N}_0 + \bot) \times (\mathrm{N}_1 + \bot)$.

The event set of net $(\mathrm{N}_0 + \bot) \times (\mathrm{N}_1 + \bot)$ contains all possible synchronisations of events in N_0 and N_1, together with an asynchronous event corresponding to each event of either N_0 or N_1.

8 A Compositional Theory for Petri Nets

The categorical constructs we have defined can be used to give an algebraic means of building large nets from smaller components in such a way that the behaviour of the composite net can be expressed in terms of the behaviour of its components.

An important technique in building complex nets from simpler ones is that of restriction, by which we restrict a net to a specified subset of its possible behaviours. For example, if we limit our attention to a subset E' of the events of a net $\mathrm{N} = \langle E, B, pre, post\rangle$ then the net with this behaviour is $\mathrm{N}' = \langle E, B, pre(\iota), post(\iota)\rangle$ where ι is the inclusion of E' in E, thus in **Set**:

$$\begin{tikzcd}
E' \times B \arrow[r, tail, "\iota \times 1"] & E \times B \arrow[r, shift left=2, "pre"] \arrow[r, shift right=2, "post"'] \arrow[r, phantom, "\times"] & \mathsf{N}.
\end{tikzcd}$$

In particular, we often restrict ourselves to a subset of the events of a product net, thus restricting the possible synchronisations of its component nets.

Thus if the event $\langle e_0, e_1\rangle \notin E'$, the events e_0 and e_1 of the component nets may not synchronise. We specify such restrictions using a synchronisation function.

Definition 8.1
A synchronisation function *on a product net* $\mathrm{N}_0 \times \mathrm{N}_1$ *is a function* $s : E_0 \times E_1 \to \{0, 1\}$.

Definition 8.2 *Let s be a synchronisation function on the net* $\mathrm{N}_0 \times \mathrm{N}_1$. *The restriction of* $\mathrm{N}_0 \times \mathrm{N}_1$ *by s, written* $\mathrm{N}_0 \times \mathrm{N}_1\lceil_s$, *is the net with event set* $E' = \{\langle e_0, e_1\rangle \in E_0 \times E_1 \mid s\langle e_0, e_1\rangle = 1\}$, *condition set* $B_0 + B_1$ *and pre- and post-condition relations given respectively by the restriction to* $E' \times (B_0 + B_1)$ *of the pre- and post-condition relations of* $\mathrm{N}_0 \times \mathrm{N}_1$.

Thus the net $\mathrm{N}_0 \times \mathrm{N}_1\lceil_s$ is such that

- e_0 and e_1 can synchronise if and only if $s\langle e_0, e_1\rangle = 1$ and $e_0, e_1 \neq *$,
- e can occur asynchronously if and only if either $s(e, *) = 1$ or $s(*, e) = 1$, and
- the idling event $\langle *, *\rangle$ is included if and only if $s\langle *, *\rangle = 1$.

Remark 8.3 *If* $* \notin (E_0 \cup E_1)$ *then no events can occur asynchronously and we do not allow idling.*

The following proposition gives a simple condition on synchronisation functions which ensures that simulation is preserved by restriction.

Proposition 8.4 Let $\langle f, F\rangle$ be a morphism in **GNet** from $\mathrm{N}_0 \times \mathrm{N}_1$ to $\mathrm{N}'_0 \times \mathrm{N}'_1$. Let s and s' be synchronisation functions on $\mathrm{N}_0 \times \mathrm{N}_1$ and $\mathrm{N}'_0 \times \mathrm{N}'_1$ respectively. The restriction of $\langle f, F\rangle$ to $\mathrm{N}_0 \times \mathrm{N}_1\lceil_s$ factors through $\mathrm{N}'_0 \times \mathrm{N}'_1\lceil_{s'}$ if and only if $s \leq s' f$.

Proof: A morphism g making the diagram:

$$\begin{tikzcd}
\mathrm{N}_0 \times \mathrm{N}_1\lceil_s \arrow[r, "g"] \arrow[d] & \mathrm{N}'_0 \times \mathrm{N}'_1\lceil_{s'} \arrow[d] \\
\mathrm{N}_0 \times \mathrm{N}_1 \arrow[r, "\langle f, F\rangle"'] & \mathrm{N}'_0 \times \mathrm{N}'_1
\end{tikzcd}$$

commute must be of form $\langle f', F\rangle$ where f' is the restriction of f to the event set of $\mathrm{N}_0 \times \mathrm{N}_1\lceil_s$. The diagram clearly commutes if and only if whenever $\langle e_0, e_1\rangle$ is an event of $\mathrm{N}_0 \times \mathrm{N}_1\lceil_s$, then $f'\langle e_0, e_1\rangle$ is an event of $\mathrm{N}'_0 \times \mathrm{N}'_1\lceil_{s'}$, which is if and only if $s'f'\langle e_0, e_1\rangle \geq s\langle e_0, e_1\rangle$ for each event $\langle e_0, e_1\rangle$ of $\mathrm{N}'_0 \times \mathrm{N}'_1\lceil_{s'}$, which is if and only if $s'f \geq s$.
$\square$

Thus $\mathrm{N}'_0 \times \mathrm{N}'_1\lceil_{s'}$ simulates $\mathrm{N}_0 \times \mathrm{N}_1\lceil_s$ if

- $\mathrm{N}'_0 \times \mathrm{N}'_1$ simulates $\mathrm{N}_0 \times \mathrm{N}_1$
- whenever e_0 and e_1 synchronise in $\mathrm{N}_0 \times \mathrm{N}_1\lceil_s$ then $\pi_0 f\langle e_0, e_1\rangle$ and $\pi_1 f\langle e_0, e_1\rangle$ synchronise in $\mathrm{N}'_0 \times \mathrm{N}'_1\lceil_{s'}$ and
- whenever $\pi_i e$ is asynchronous in $\mathrm{N}_0 \times \mathrm{N}_1\lceil_s$ then $\pi_i f e$ is asynchronous in $\mathrm{N}'_1\lceil_{s'}$.

Note that the presence of the trivial event $\langle *, *\rangle$ is not preserved by morphisms in **GNet**.

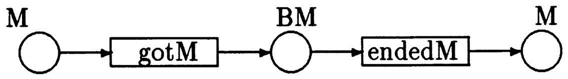
Proposition 8.5 Let s be a synchronisation function on N such that $\mathrm{N}\lceil_s$ has event set E'. Let A be a multiset over E' and let $\widehat{A}$ be the multiset over E such that $\widehat{A}(e) = A(e)$ for $e \in E'$ and $\widehat{A}(e) = 0$ otherwise. Let C be a multiset over E, and $\overline{C}$ the restriction of C to E'. Then

- if $M \rightsquigarrow M'$ in $\mathrm{N}\lceil_s$ under A, then $M \rightsquigarrow M'$ in N under $\widehat{A}$, and
- if $C(e) = 0$ for all $e \in (E \setminus E')$ then whenever $M \rightsquigarrow^{C} M'$ in N, we have $M \rightsquigarrow^{\overline{C}} M'$.

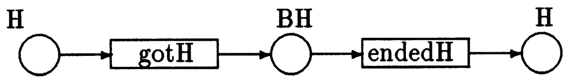
Proof: Straightforward. $\square$

Since every evolution of the restricted net is an evolution of the unrestricted net, restriction preserves safety properties (such as deadlock avoidance) but not liveness properties (such as reachability).

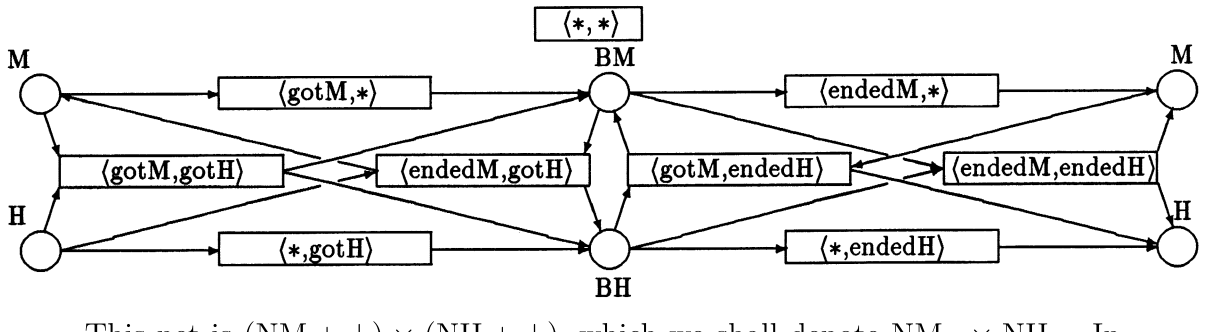
We illustrate the use of product, coproduct and restriction using an extended example related to the *Jobshop* example of [Mil89]p25. Our intention is to demonstrate the algebraic nature of our approach, rather than to make a direct analogy with CCS. We shall also illustrate the interaction of simulation morphisms with net constructors. It is an immediate consequence of the functoriality of the constructions $\times, +, \otimes$ and $\multimap$ that if for $i \in \{1, \ldots, n\}$ the net N_i' simulates N_i, then $op(\mathrm{N}_1', \ldots, \mathrm{N}_n')$ simulates $op(\mathrm{N}_1, \ldots, \mathrm{N}_n)$ where op may be any of $\wedge, \oplus, \otimes$ and $\multimap$. The interaction of simulation morphisms with restriction is given by Proposition 8.4. We shall specify a net representing a *Jobshop* by composing several smaller component nets. We first specify the behaviour of a mallet (M) (drawing the condition M twice):

NM =


Thus, if a mallet is "got", it becomes a busy mallet (BM), and when its use is ended, it returns to its original state. It is readily proved (if not obvious) that the number of tokens on the net NM remains constant through out any behaviour of the net. It is the number of mallets in the jobshop. The specification of a hammer (H) is similar:

NH =


The behaviour of mallets and hammers together, or alone, might be described by the following net:



This net is $(\mathrm{NM} + \perp) \times (\mathrm{NH} + \perp)$, which we shall denote $\mathrm{NM}_\perp \times \mathrm{NH}_\perp$. In fact, we do not wish to force the synchronised use of mallets and hammers.

To do so would imply either that each was used for the same length of time and never alone, or else that one was unnecessarily barred from use until the other became free. Similarly, there is no reason to synchronise the getting of a hammer with ending the use of a mallet. Therefore we shall restrict on the synchronised events, to give a net representing the asynchronous use of hammers and mallets.

We achieve this using the synchronisation function s given by

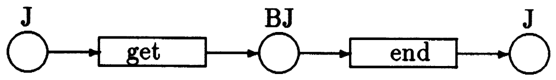
$$s\langle \text{gotM}, *\rangle = s(*, \text{gotH}\rangle = s\langle \text{endedM}, *\rangle = s\langle *, \text{endedH}\rangle = 1 \text{ and}$$
$$s\langle e_0, e_1\rangle = 0 \text{ otherwise.}$$

Then the restricted net $(\text{NM}_\perp \times \text{NH}_\perp)\lceil_s$ is such that

- no two events can synchronise
- every event of NM and NH can occur asynchronously, and
- the trivial event is not included.

The construction $(\text{NM}_\perp \times \text{NH}_\perp)\lceil_s$ is comparable with the CCS parallel operator $|$. In CCS we restrict parallel compositions by requiring certain actions to occur only in synchronisation with specified complementary actions from separate processes. Here, we restrict to specified synchronisations of events from each component process and to specified asynchronous events.

We next specify a jobber (J) in terms of the use of tools:

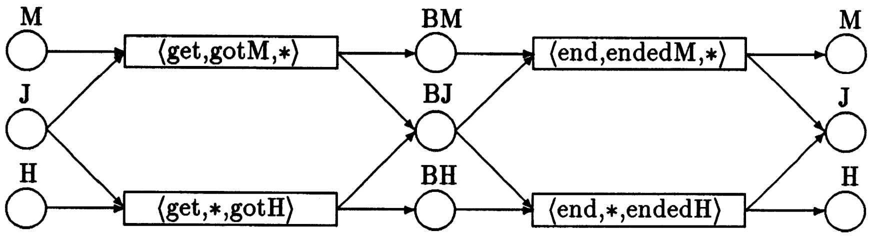
NJ = 

The jobber gets a tool, becoming busy (BJ). After using the tool, the jobber resumes ner initial state. The total number of tokens on this net is constant throughout any behaviour, and is the number of jobbers in our jobshop.

Consider the product net $\text{NJ} \times (\text{NM}_\perp \times \text{NH}_\perp\lceil_s)$. It has events:

$\langle$get, gotM, $*\rangle$	$\langle$get, $*$, gotH$\rangle$	$\langle$end, endedM, $*\rangle$	$\langle$end, $*$, endedH$\rangle$
$\langle$get, endedM, $*\rangle$	$\langle$get, $*$, endedH$\rangle$	$\langle$end, gotM, $*\rangle$	$\langle$end, $*$, gotH$\rangle$

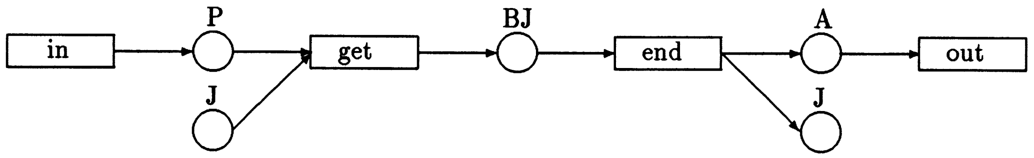
We want only the first four of these events, imposing the reasonable requirement that a jobber cannot get a tool unless a tool is simultaneously got, and cannot end the use of a tool unless the use of a tool is ended. Restricting to these four events, using the appropriate synchronisation function s', we obtain the net:



$\mathbf{NJ} \times (\mathbf{NM}_\perp \times \mathbf{NH}_\perp\lceil_s)\lceil_{s'}$

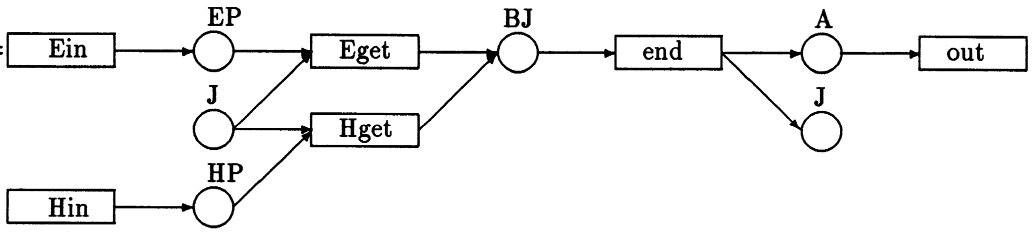
Which of ⟨get, gotM, ∗⟩ and ⟨get, ∗, gotH⟩ occurs may be determined by the environment, since either H or M may be marked. If both are marked, a non-deterministic choice is made internally by the system.

We can describe further interaction with the outside world, as Milner does, by considering how the jobber receives component pieces (P) and outputs completed jobs (A), by specifying a new jobber thus:

NJ′ = 

Observe that this net can be thought of as a restriction of the product of NJ with a net specifying the behaviour of pieces. If we form the product $\mathrm{NJ}'_\perp \times (\mathrm{NM}_\perp \times \mathrm{NH}_\perp\lceil_s)$ and restrict appropriately (by a synchronisation function we shall denote t), we obtain a net describing a jobber's behaviour with respect to hammers and mallets as before, but including also the interaction between the jobber and the pieces and completed jobs.

Now consider the following net, which assumes that pieces may constitute an easy job (EP) or a hard job (HP):

NJob = 

Now, NJob simulates NJ$'$, since there is a morphism $\langle f, F\rangle$ in **GNet** from NJob to NJ$'$ given by

$f(\text{get}) = \text{Eget}$	$f(\text{end}) = \text{end}$		
$F(\text{EP}) = F(\text{HP}) = P$	$F(\text{J}) = \text{J}$	$F(\text{BJ}) = \text{BJ}$	$F(\text{A}) = \text{A}.$

We shall assume that the jobber must use a mallet to assemble the hard pieces, but that either a hammer or a mallet may be used to assemble the easy pieces. With this requirement in mind, we restrict the product net NJob $\times$ (NM$_\perp \times$ NH$_\perp\lceil_s$) by a synchronisation function we shall denote t', to obtain the net with events:

$\langle$Eget, $*$, gotH$\rangle$	$\langle$Eget, gotH, $*\rangle$	$\langle$Hget, gotM, $*\rangle$	$\langle$end, endedH, $*\rangle$
$\langle$end, $*$, endedH $\rangle$	$\langle$Ein, $*, *\rangle$	$\langle$Hin, $*, *\rangle$	$\langle$out, $*, *\rangle$

and pre- and post-condition relations given by:

pre$\langle$Eget,$*$,gotH$\rangle$ = EP+H+J	*post*$\langle$Eget,$*$,gotH$\rangle$ = BJ+BH	*pre*$\langle$Ein,$*, *\rangle = \emptyset$
pre$\langle$Eget,gotM,$*\rangle$ = EP+M+J	*post*$\langle$Eget,gotM,$*\rangle$ = BJ+BM	*post*$\langle$Ein,$*, *\rangle$ = EP
pre$\langle$Hget,gotM,$*\rangle$ = HP+M+J	*post*$\langle$Hget,gotM,$*\rangle$ = BJ+BM	*pre*$\langle$Hin,$*, *\rangle = \emptyset$
pre$\langle$end,endedM,$*\rangle$ = BJ+BM	*post*$\langle$end,endedM,$*\rangle$ = M+J+A	*post*$\langle$Hin,$*, *\rangle$ = HP
pre$\langle$end,$*$,endedH$\rangle$ = BJ+BM	*post*$\langle$end,$*$,endedH$\rangle$ = H+J+A	*pre*$\langle$out,$*, *\rangle$ = A

Now we have a morphism $\langle f, F\rangle \times 1$ in **GNet** from NJ$'$ $\times$ (NM$_\perp \times$ NH$_\perp\lceil_s$) to NJob $\times$ (NM$_\perp \times$ NH$_\perp\lceil_s$). Further, t and t' satisfy the conditions of Lemma 8.4, and so $\langle f, F\rangle \times 1$ restricts to a morphism from NJ$'$ $\times$ (NM$_\perp \times$ NH$_\perp\lceil_s)\lceil_t$ to NJob $\times$ (NM$_\perp \times$ NH$_\perp\lceil_s)\lceil_{t'}$. Thus we have a simulation of the compound net NJ$'$ $\times$ (NM$_\perp \times$ NH$_\perp\lceil_s)\lceil_t$ by a compound net which has been extended to consider easy and hard jobs.

8.1 The exponential of a net

In [DP91] de P aiva modelled the linear modality "of course" by constructing a comonad ! with the properties that,

$$!A\otimes !B = !(A \times B) \text{ and } !1 = \text{I}$$

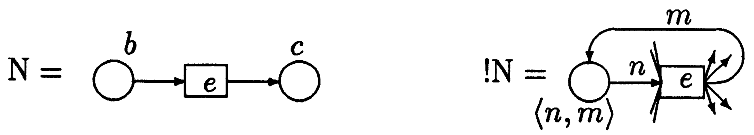
By Theorem 7.3, this comonad lifts to a comonad ! in **GNet** with the same properties. It has the following definition.

Definition 8.6 *The* exponential !N *of the net* N *is the net* $\langle E, (B^*)^E, pre_!, post_! \rangle$ *where* B^* *is the free commutative monoid ove* B, *and if* $\phi : E \to B^*$ *is the function given by* $\phi(e) = \sum_B m_i b_i$ *then*

$$pre_!(e, \phi) = \sum_B m_i pre(e, b_i) \quad \text{and} \quad post_!(e, \phi) = \sum_B m_i post(e, b_i).$$

The net !N has the same event set as N but a considerably larger condition set. We may understand the net !N as a net in which the conditions have been "unwound" into all their possible requirements. This may be seen in the following example.

Example 8.7



The net !N has one event e and countably many conditions, labelled by pairs (because B has *two* elements) of integers. The condition $\langle n, m \rangle$ appears in $pre(e)$ with multiplicity n and in $post(e)$ with multiplicity m.

8.2 The symmetric monoidal closed structure of GNet

GNet has a symmetric monoidal closed structure induced by that in $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$. The tensor product $\otimes$ and its right adjoint, linear implication ($\multimap$), have the following definitions.

Definition 8.8 *The* tensor product $(N_0 \otimes N_1)$ *of two nets is the net* $\langle E_0 \times E_1, B_0^{E_1} \times B_1^{E_0}, pre_0 \otimes pre_1, post_0 \otimes post_1 \rangle$ *where,*

$$pre_0 \otimes pre_1 \langle e_0, e_1, f, g \rangle = pre_0 \langle e_0, f e_1 \rangle + pre_1 \langle e_1, g e_0 \rangle$$

$$post_0 \otimes post_1 \langle e_0, e_1, f, g \rangle = post_0 \langle e_0, f e_1 \rangle + post_1 \langle e_1, g e_0 \rangle$$

Its unit is the net $\mathrm{I} = \langle \{*\}, \{*\}, 0, 0 \rangle$.

Definition 8.9 *The* linear implication $(\mathrm{N}_0 \multimap \mathrm{N}_1)$ *of two nets is the net* $\langle E_1^{E_0} \times B_0^{B_1}, E_0 \times B_1, pre_0 \multimap pre_1, post_0 \multimap post_1 \rangle$ *where,*

$$(pre_0 \multimap pre_1)\langle f, F, e_0, b_1 \rangle = pre_0 \langle f e_0, b_1 \rangle \ominus pre_1 \langle e_0, F b_1 \rangle$$

$$(post_0 \multimap post_1)\langle f, F, e_0, b_1 \rangle = post_0 \langle f e_0, b_1 \rangle \ominus post_1 \langle e_0, F b_1 \rangle$$

and $\ominus$ *is the truncated subtraction described in Appendix A.2.*

9 A Proof System for Nets

In Section 8 we made implicit use of a language for nets. In this section, we shall describe how to extend this to a proof system for nets, based on Girard's intuitionistic linear logic [Gir87]. We first define a language $\mathcal{N}$ for nets by the following BNF,

$$\mathcal{N} ::= 0 \mid 1 \mid \mathrm{I} \mid \mathrm{N}_i \mid \mathrm{N} \oplus \mathrm{N}' \mid \mathrm{N} \wedge \mathrm{N}' \mid \mathrm{N} \otimes \mathrm{N}' \mid \mathrm{N} \multimap \mathrm{N}' \mid !\mathrm{N} \mid \mathrm{N}\lceil_s$$

where the N_i and s are chosen from two disjoint collections of constant symbols.
We shall interpet this language in **GNet**. The interpretation is parametric in a function τ which assigns an object of **GNet**, that is a Petri net, to each of the constant symbols N_i and a synchronisation function to each synchronisation symbol s. Each term of the language $\mathcal{N}$ is then interpreted by an object of **GNet** in the following way.

- $[\![\mathrm{N}_i]\!] = \tau(\mathrm{N}_i)$ for each constant symbol N_i,
- $[\![\mathbf{1}]\!] = \mathbf{1} = \langle \{*\}, \phi, \phi, \phi \rangle$,
- $[\![\mathbf{0}]\!] = \mathbf{0} = \langle \phi, \{*\}, \phi, \phi \rangle$,
- $[\![\mathrm{I}]\!] = \mathrm{I} = \langle \{*\}, \{*\}, 0, 0 \rangle$,
- $[\![\mathrm{N} \wedge \mathrm{N}']\!] = [\![\mathrm{N}]\!] \times [\![\mathrm{N}']\!]$,
- $[\![\mathrm{N} \oplus \mathrm{N}']\!] = [\![\mathrm{N}]\!] + [\![\mathrm{N}']\!]$,

- $[\![\mathrm{N} \otimes \mathrm{N}']\!] = [\![\mathrm{N}]\!] \otimes [\![\mathrm{N}']\!]$,
- $[\![\mathrm{N} \multimap \mathrm{N}']\!] = [\![\mathrm{N}]\!] \multimap [\![\mathrm{N}']\!]$,
- $[\![!\mathrm{N}]\!] = ![\![\mathrm{N}]\!]$ and
- $[\![\mathrm{N}\lceil_s]\!] = [\![\mathrm{N}]\!]\lceil_{\tau(s)}$

With the exception of restriction, our language contains just the connectives of intuitionistic linear logic (LIL). Therefore, the evident choice is to use the following rules of intuitionistic linear logic in order to derive net refinements.

$$\frac{}{\mathrm{N} \vdash \mathrm{N}}\ (\mathrm{Id}) \qquad \frac{\Gamma \vdash \mathrm{N} \quad \mathrm{N}, \Delta \vdash \mathrm{N}'}{\Gamma, \Delta \vdash \mathrm{N}'}\ (\mathrm{Cut}) \qquad \frac{\Gamma, \mathrm{N}, \mathrm{N}' \vdash \mathrm{N}''}{\Gamma, \mathrm{N}', \mathrm{N} \vdash \mathrm{N}''}\ (\mathrm{Ex})$$

$$\frac{}{\Gamma, \mathbf{0} \vdash \mathrm{N}}\ (\mathbf{0} \vdash) \qquad \frac{}{\Gamma \vdash \mathbf{1}}\ (\vdash \mathbf{1}) \qquad \frac{\Gamma \vdash \mathrm{N}}{\Gamma, \mathrm{I} \vdash \mathrm{N}}\ (\mathrm{I} \vdash) \qquad \frac{}{\vdash \mathrm{I}}\ (\vdash \mathrm{I})$$

$$\frac{\Gamma \vdash \mathrm{N} \quad \Delta \vdash \mathrm{N}'}{\Gamma, \Delta \vdash \mathrm{N} \otimes \mathrm{N}'}\ (\vdash \otimes) \qquad \frac{\Gamma, \mathrm{N}, \mathrm{N}' \vdash \mathrm{N}''}{\Gamma, \mathrm{N} \otimes \mathrm{N}' \vdash \mathrm{N}''}\ (\otimes \vdash)$$

$$\frac{\Gamma \vdash N}{\Gamma \vdash N \oplus N'}\ (\vdash \oplus \mathrm{l}) \qquad \frac{\Gamma \vdash N'}{\Gamma \vdash \mathrm{N} \oplus \mathrm{N}'}\ (\otimes \vdash\ \mathrm{r})$$

$$\frac{\Gamma, \mathrm{N} \vdash \mathrm{N}'' \quad \Gamma, \mathrm{N}' \vdash \mathrm{N}''}{\Gamma, \mathrm{N} \oplus \mathrm{N}' \vdash \mathrm{N}''}\ (\otimes \vdash) \qquad \frac{\Gamma, \mathrm{N} \quad \Gamma \vdash \mathrm{N}'}{\Gamma \vdash \mathrm{N} \wedge \mathrm{N}'}\ (\vdash \wedge)$$

$$\frac{\Gamma, \mathrm{N} \vdash \mathrm{N}''}{\Gamma, \mathrm{N} \wedge \mathrm{N}' \vdash \mathrm{N}''}\ (\mathrm{l} \wedge \vdash) \qquad \frac{\Gamma, \mathrm{N}' \vdash \mathrm{N}''}{\Gamma, \mathrm{N} \wedge \mathrm{N}' \vdash \mathrm{N}''}\ (\mathrm{r} \wedge \vdash)$$

$$\frac{\Gamma \vdash \mathrm{N} \quad \mathrm{N}' \vdash \mathrm{N}''}{\Gamma, \mathrm{N} \multimap \mathrm{N}' \vdash \mathrm{N}''}\ (\multimap\vdash) \qquad \frac{\Gamma, \mathrm{N} \vdash \mathrm{N}'}{\Gamma \vdash \mathrm{N} \multimap \mathrm{N}'}\ (\vdash\multimap)$$

$$\overline{!\mathrm{N} \vdash \mathrm{I} \wedge \mathrm{N} \wedge (!\mathrm{N} \otimes !\mathrm{N})} \ (! \multimap)$$

The following result, which is an immediate consequence of Theorem 7.3, is our justification for using this as a basis of a proof system for nets.

Proposition 9.1 Let $[\![-]\!]$ be an interpretation of $\mathcal{N}$ in **GNet**. If $\Gamma \vdash \mathrm{N}$ then there is a morphism from $[\![\Gamma]\!]$ to $[\![\mathrm{N}]\!]$.

The existence of a derivation $\mathrm{N} \vdash \mathrm{N}'$ implies the existence of a morphism from $[\![\mathrm{N}]\!]$ to $[\![\mathrm{N}']\!]$ in **GNet**, and hence that N' simulates N.

We would like to extend our system to deal with restriction by adding a rule of the form:

$$\frac{\mathrm{N} \vdash \mathrm{N}'}{\mathrm{N}\lceil_s \vdash \mathrm{N}'\lceil_{s'}}$$

together with a suitable side condition on s and s'. Unfortunately, it follows from Propsition 8.4 that the side condition will depend on the choice of morphism between N' and N. The derivability of $\mathrm{N} \vdash \mathrm{N}'$ implies only that there is a morphism from N' to N.

For this reason, it seems likely that it would be profitable to develop a proof system based on Girard and Lafont's term assignment system for intuitionistic linear logic [GL87], in which one derives sequents of the form $x : \Gamma \vdash t : A$. Our rule for restriction would then be of the form:

$$\frac{x : \mathrm{N} \vdash t : \mathrm{N}'}{x : \mathrm{N}\lceil_s \vdash t : \mathrm{N}'\lceil_{s'}}$$

with the side condition that $s \leq s't$.

This appears to be a promising direction for future investigation, and is currently work in progress.

10 Marked Nets

In this paper, as in [Bro90, BG90, BG], we have defined categories of nets without an initial marking. Other authors ([Win87], [MM88a]) have constructed categories of marked nets. An important consequence of the results of Section 6 is that they provide a clear indication as to how to extend our work to construct a category of marked nets.

Lemma 10.1 The following data:

- objects: marked nets $\langle \mathrm{N}, M \rangle$,
- morphisms: pairs $\langle f, F \rangle$ such that $\langle f, F \rangle$ is a morphism from N to N$'$ in **GNet** and $\langle M, M' \rangle$ is F-`ok`, and
- composition: as in **GNet**

define a category, **MNet**.

> **Proof:** It is clear that if $\langle \mathrm{N}, M \rangle$ is a marked net then $\langle M, M \rangle$ is *id* `-ok` and so we have identities. Composition is well-defined since if $\langle M_0, M_1 \rangle$ is F-`ok` and $\langle M_1, M_2 \rangle$ is G-`ok` then $M_0 F \leq M_1$ and $M_1 G \leq M_2$ and so $M_0 FG \leq M_2$ whence $\langle M_0, M_1 \rangle$ is FG-`ok`. Associativity is inherited from **GNet**. □

There is an evident forgetful functor $\mathcal{U}$: **MNet** → **GNet** mapping a marked net $\langle \mathrm{N}, M \rangle$ to N and a morphism $\langle f, F \rangle$ to $\langle f, F \rangle$.

Lemma 10.2 The assignment:

$$\mathrm{N} \longmapsto \langle \mathrm{N}, 0 \rangle \quad \text{and} \quad \langle f, F \rangle \longmapsto \langle f, F \rangle,$$

where 0 is the constant zero function, defines a full and faithful functor $\mathcal{F}$: **GNet** → **MNet**.

> **Proof:** $\mathcal{F}$ is well-defined since $\langle 0, 0 \rangle$ is F-`ok` for any F. $\mathcal{F}$ is clearly faithful, and is full since every map from $\langle \mathrm{N}, 0 \rangle$ to $\langle \mathrm{N}', 0 \rangle$ in **MNet** is a map from N to N$'$ in **GNet**. □

Proposition 10.3 $\mathcal{F}$ is left adjoint to $\mathcal{U}$

Proof: $$\begin{aligned}\mathbf{MNet}(\mathcal{F}\mathrm{N}, \langle \mathrm{N}', M'\rangle) &= \mathbf{MNet}(\langle \mathrm{N}, 0\rangle, \langle \mathrm{N}', M'\rangle)\\ &= \{\langle f, F\rangle \in \mathbf{GNet}(\mathrm{N}, \mathrm{N}') \mid \langle 0, M'\rangle \\ &\quad \text{is } F\text{-}\mathtt{ok}\}\\ &= \mathbf{GNet}(\mathrm{N}, \mathrm{N}') \text{ since } \langle 0, M'\rangle\\ &\quad \text{is } F\text{-}\mathtt{ok} \text{ for any } F, M'\\ &= \mathbf{GNet}(\mathrm{N}, \mathcal{U}\langle \mathrm{N}', M'\rangle).\end{aligned}$$
as required. □

If we extend the definition of a marking to a function $M : B \to \mathbb{N} \cup \{\omega\}$ then we can also construct a right adjoint to $\mathcal{U}$.

Lemma 10.4 The assignment:

$$\mathrm{N} \longmapsto \langle \mathrm{N}, \omega\rangle \quad \text{and} \quad \langle f, F\rangle \longmapsto \langle f, F\rangle,$$

where ω is the constant ω function, defines a full and faithful functor $\mathcal{G}$: **GNet** $\to$ **MNet**.

Proof: $\mathcal{G}$ is well-defined since $\langle \omega, \omega\rangle$ is F-`ok` for any F. $\mathcal{G}$ is clearly faithful, and is full since every map from $\langle \mathrm{N}, \omega\rangle$ to $\langle \mathrm{N}', \omega\rangle$ in **MNet** is a map from N to N$'$ in **GNet**. □

Proposition 10.5 $\mathcal{G}$ is right adjoint to $\mathcal{U}$.

Proof: $$\begin{aligned}\mathbf{MNet}(\langle \mathrm{N}, M\rangle, \mathcal{G}\mathrm{N}') &= \mathbf{MNet}\ (\langle \mathrm{N}, M\rangle, \langle \mathrm{N}, \omega\rangle)\\ &= \{\langle f, F\rangle \in \mathbf{GNet}(\mathrm{N}, \mathrm{N}') \mid \langle M, \omega\rangle\\ &\quad \text{is } F\text{-}\mathtt{ok}\}\\ &= \mathbf{GNet}(\mathrm{N}, \mathrm{N}') \text{ since } \langle M, \omega\rangle\\ &\quad \text{is } F\text{-}\mathtt{ok} \text{ for any } F, M'\\ &= \mathbf{GNet}(\mathcal{U}\langle \mathrm{N}', M'\rangle, \mathrm{N}').\end{aligned}$$
as required. □

Corollary 10.6 GNet is isomorphic to both a full reflective and a full coreflective subcategory of **MNet**.

Applying the adjoint functor theorem[5] shows that $\mathcal{U}$ preserves any (small)

[5]The solution set condition is trivial since any morphism $\langle f, F\rangle : \mathrm{N} \to \mathcal{U}\mathrm{N}'$ equals $\mathcal{U}(\langle f, F\rangle : \mathcal{F}\mathrm{N} \to \mathrm{N}')$.

limits and colimits that exist in **MNet**. Thus, limits and colimits in **MNet** can differ from those in **GNet** only in their action on markings. In particular, we have the following results.

Proposition 10.7 MNet has binary products given by,

$$\langle \mathrm{N}_0, M_0\rangle \times \langle \mathrm{N}_1, M_1\rangle = \langle \mathrm{N}_0 \times \mathrm{N}_1, M_0 \wedge M_1\rangle$$

where $\mathrm{N}_0 \times \mathrm{N}_1$ is the product of N_0 and N_1 in **GNet**, and $M_0 \wedge M_1 : B_0 + B_1 \to \mathbb{N}$ is given by $M_0 \wedge M_1(in_i(b)) = M_i(b)$.

> **Proof:** Projections in **GNet** are given by $\langle \pi_i, in_i\rangle : \langle \mathrm{N}_0 \times \mathrm{N}_1\rangle \to \mathrm{N}_i$. $\langle \pi_i, in_i\rangle$ is a morphism from $\langle \mathrm{N}_0, M_0\rangle \times \langle \mathrm{N}_1, M_1\rangle$ to $\langle \mathrm{N}_i, M_i\rangle$ as $(\langle M_0 \wedge M_1\rangle) in_i = M_i \leq M_i$ and so $\langle M_0 \wedge M_1, M_i\rangle$ is in_i-`ok`.
>
> Now suppose that there exists $\langle f_i, F_i\rangle : \langle \mathrm{N}, M\rangle \to \langle \mathrm{N}_i, M_i\rangle$. Then there exists a unique map $\langle\langle f_0, f_1\rangle, F_0 + F_1\rangle : \mathrm{N} \to \mathrm{N}_0 \times \mathrm{N}_1$ in **GNet** such that $\langle \pi_i, in_i\rangle\langle\langle f_0, f_1\rangle, F_0 + F_1\rangle = \langle f_i, F_i\rangle$.
> However, the $\langle f_i, F_i\rangle$ are morphisms in **MNet** and so each $\langle M, M_i\rangle$ is F_i-`ok`. Therefore, $M(F_0 + F_1)(in_i(b)) = MF_i(b) \leq M_i(b) = (M_0 \wedge M_1) in_i(b)$ and so $\langle M, M_0 \wedge M_1\rangle$ is $(F_0 + F_1)$-`ok` whence $\langle\langle f_0, f_1\rangle, F_0 + F_1\rangle$ is a morphism in **MNet**. This completes the proof. □

Proposition 10.8 MNet has binary coproducts given by,

$$\langle \mathrm{N}_0, M_0\rangle + \langle \mathrm{N}_1, M_1\rangle = \langle \mathrm{N}_0 + \mathrm{N}_1, M_0 \oplus M_1\rangle$$

where $\mathrm{N}_0 + \mathrm{N}_1$ is the coproduct of N_0 and N_1 in **GNet**, and $M_0 \oplus M_1 : B_0 \times B_1 \to \mathbb{N}$ is given by $M_0 \oplus M_1\langle b_0, b_1\rangle = \max\{M_0(b_0), M_1(b_1)\}$.

> **Proof:** Injections in **GNet** are given by $\langle in_i, \pi_i\rangle : \mathrm{N}_i \to \mathrm{N}_0 + \mathrm{N}_1$. $\langle in_i, \pi_i\rangle$ is a morphism from $\langle \mathrm{N}_i, M_i\rangle$ to $\langle \mathrm{N}_0, M_0\rangle + \langle \mathrm{N}_1, M_1\rangle$ as $M_0 \oplus M_1(\langle b_0, b_1\rangle) = \max\{M_0(b_0), M_1(b_1)\} \geq M_i(b_i) = M_i\pi_i(\langle b_0, b_1\rangle)$ and so $\langle M_i, M_0 \oplus M_1\rangle$ is π_i-`ok`.
> Now suppose that there exists $\langle f_i, F_i\rangle : \langle \mathrm{N}_i, M_i\rangle \to \langle \mathrm{N}, M\rangle$. Then there exists a unique map $\langle f_0 + f_1, \langle F_0, F_1\rangle\rangle : \mathrm{N}_0 + \mathrm{N}_1 \to \mathrm{N}$ in **GNet** such that $\langle f_0 + f_1, \langle F_0, F_1\rangle\rangle\langle in_i, \pi_i\rangle = \langle f_i, F_i\rangle$.
> However, the $\langle f_i, F_i\rangle$ are morphisms in **MNet** and so each $\langle M_i, M\rangle$

is F_i-`ok`. Therefore, $M_0 \oplus M_1\langle F_0, F_1\rangle(b) = M_0 \oplus M_1\langle F_0 b, F_1 b\rangle = \max\{M_0 F_0 b, M_1 F_1 b\} \leq M(b)$ as $M(b)$ is greater than each $M_i F_i(b)$ since, each $\langle M_i, M\rangle$ is F-`ok`. Thus $\langle f_0{+}f_1, \langle F_0, F_1\rangle\rangle$ is a morphism in **MNet**. This completes the proof. $\square$

Proposition 10.9 MNet has initial and terminal objects given by,

$$0_{\mathbf{MNet}} = \langle 0_{\mathbf{GNet}}, 0\rangle \quad \text{and} \quad 1_{\mathbf{MNet}} = \langle 1_{\mathbf{GNet}}, \emptyset\rangle.$$

Proof: There exists a unique $*$ from 0 to N in **GNet**. This is also a map from 0 to $\langle \mathrm{N}, M\rangle$ in **MNet** for any M since $M \geq 0* = 0$. There exists a unique $*$ from N to 1 in **GNet**. This is also a map from $\langle \mathrm{N}, M\rangle$ to 1 in **MNet** for any M since $\emptyset \geq M\emptyset = \emptyset$. $\square$

The symmetric monoidal closed structure of **GNet** does not lift to **MNet** as smoothly as we might have hoped. Recall from Section 8 that the net $\mathrm{N}_0 \otimes \mathrm{N}_1$ is given by,

$$\langle E_0 \times E_1, B_0^{E_1} \times B_1^{E_0}, \mathit{pre}_0 \otimes \mathit{pre}_1, \mathit{post}_0 \otimes \mathit{post}_1\rangle.$$

Therefore, a marking of the tensor product of marked nets $\langle \mathrm{N}_0, M_0\rangle$ and $\langle \mathrm{N}_1, M_1\rangle$ is a function $M_0 \otimes M_1$ from $B_0^{E_1} \times B_1^{E_0}$ to $\mathbb{N}$. The natural choice appears to be to define,

$$M_0 \otimes M_1\langle f, g\rangle = \max_{E_0}\{M_1 f(e_0)\} + \max_{E_1}\{M_0 g(e_1)\}.$$

This has the appealing property that if $M_0 \downarrow e_0$ and $M_1 \downarrow e_1$ then $M_0 \otimes M_1 \downarrow \langle e_0, e_1\rangle$. Unfortunately, with this definition of $M_0 \otimes M_1$, it is not possible to give a definition of $M_0 \multimap M_1$ in such a way that $- \otimes \langle \mathrm{N}_0, M_0\rangle$ is left adjoint to $- \multimap \langle \mathrm{N}_0, M_0\rangle$. The difficulty is that a marking of the net $\mathrm{N}_0 \multimap \mathrm{N}_1$ which is,

$$\langle E_1^{E_0} \times B_0^{B_1}, E_0 \times B_1, \mathit{pre}_0 \multimap \mathit{pre}_1, \mathit{post}_0 \multimap \mathit{post}_1\rangle.$$

is a function $M_0 \multimap M_1$ from $E_0 \times B_1$ to $\mathbb{N}$. For the adjunction we require,

$$\begin{aligned} \forall b_2 \in B_2.\ M_2(b_2) \geq M_0 \otimes M_1\langle f, g\rangle \quad \text{iff} \quad & \forall b_2 \in B_2\ \forall e_1 \in E_1. \\ & M_1 \multimap M_2\langle e_1, b_2\rangle \geq M_0(f(e_1)). \end{aligned}$$

As g occurs free on the left hand side of this equivalence and does not occur on the right, the definition of $M_0 \otimes M_1$ can not depend on g. In fact, if we ignore g and define,

$$M_0 \otimes M_1\langle f, g\rangle = \max_{E_0}\{M_1 f(e_0)\}.$$

The we can recover the closed structure by defining,

$$M_0 \multimap M_1\langle e_1, b_0\rangle = M_1(b_1)$$

in which case $- \otimes \langle \mathrm{N}, M\rangle$ is left adjoint to $- \multimap \langle \mathrm{N}, M\rangle$. However, this is unsatisfactory as it destroys the symmetry of $\otimes$ and has no natural computational interpretation. These difficulties have led us to explore other possible closed structures on **GNet** and **MNet** such as those described in [LS91]. This is work in progress.

A Structure in $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$

A.1 Products and Coproducts in $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$

The products and coproducts in $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$ are defined in terms of the product and coproduct in **Set** (observe that, since products and coproducts in **Set** are *assigned*, they are also assigned in $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$). It is routine to prove the following lemmas.

Lemma A.1 The product of two objects $\langle E_0, B_0, \alpha_0\rangle$ and $\langle E_1, B_1, \alpha_1\rangle$ of $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$ is $\langle E_0 \times E_1, B_0 + B_1, \alpha_0 \wedge \alpha_1\rangle$, where the function $\alpha_0 \wedge \alpha_1$ from $(E_0 \times E_1) \times (B_0 + B_1)$ into $\mathbb{N}$ is given by

$$(\alpha_0 \wedge \alpha_1)\langle e_0, e_1, b\rangle = \begin{cases} \alpha_0\langle e_0, b_0\rangle & \text{if } b = (b_0, 0) \\ \alpha_1\langle e_1, b_1\rangle & \text{if } b = (b_1, 1). \end{cases}$$

The projection from $\langle E_0 \times E_1, B_0 + B_1, \alpha_0 \wedge \alpha_1\rangle$ to $\langle E_i, B_i, \alpha_i\rangle$ is the morphism $\langle \pi_i, in_i\rangle$, where π_i is the ith projection in **Set** and in_i the ith canonical injection in **Set**.

Lemma A.2 The coproduct of two objects $\langle E_0, B_0, \alpha_0\rangle$ and $\langle E_1, B_1, \alpha_1\rangle$

of $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$ is $\langle E_0 + E_1, B_0 \times B_1, \alpha_0 \oplus \alpha_1\rangle$ where the function $\alpha_0 \oplus \alpha_1$ from $(E_0 + E_1) \times (B_0 \times B_1)$ into $\mathbb{N}$ is given by

$$(\alpha_0 \oplus \alpha_1)\langle e, b_0, b_1\rangle = \begin{cases} \alpha_0\langle e_0, b_0\rangle & \text{if } e = (e_0, 0) \\ \alpha_1\langle e_1, b_1\rangle & \text{if } e = (e_1, 1). \end{cases}$$

The injection from $\langle E_i, B_i, \alpha_i\rangle$ into $\langle E_0 + E_1, B_0 \times B_1, \alpha_0 \oplus \alpha_1\rangle$ is the morphism $\langle \pi_i, in_i\rangle$.

A.2 A Monoidal Closed Structure on $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$

Lemma A.3 The functor $\otimes : \mathbf{M}_{\mathbb{N}}\mathbf{Set} \times \mathbf{M}_{\mathbb{N}}\mathbf{Set} \to \mathbf{M}_{\mathbb{N}}\mathbf{Set}$ which takes the pair $\langle\langle E_0, B_0, \alpha_0\rangle, \langle E_1, B_1, \alpha_1\rangle\rangle$ of objects of $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$ to

$$\langle E_0 \times E_1, B_0^{E_1} \times B_1^{E_0}, \alpha_0 \otimes \alpha_1\rangle,$$

where $(\alpha_0 \otimes \alpha_1)\langle e_0, e_1, f, g\rangle = \alpha_0\langle e_0, fe_1\rangle + \alpha_1\langle e_1, ge_0\rangle$, gives rise to a symmetric monoidal structure on $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$ (also denoted $\otimes$) with unit $I = \langle\{*\}, \{*\}, 0\rangle$.

Lemma A.4 The functor $(- \otimes \langle E_0, B_0, \alpha_0\rangle)$ has a right adjoint $(- \multimap \langle E_0, B_0, \alpha_0\rangle)$ which takes the object $\langle E_1, B_1, \alpha_1\rangle$ of $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$ to the object

$$\langle E_1^{E_0} \times B_0^{B_1}, E_0 \times B_1, \alpha_0 \multimap \alpha_1\rangle,$$

where $(\alpha_0 \multimap \alpha_1)\langle f, F, e_0, b_1\rangle = \alpha_1(fe_0, b_1 \ominus \alpha_0(e_0, Fb_1)$.

The symbol $\ominus$ represents truncated subtraction, which coincides with natural number subtraction unless its first argument is smaller than its second, in which case it evaluates to 0. This is analogous to a construction used by Lawvere on the positive reals, see [Law73].

B Products in GNet

Recall, from Section 6, that we identify a Petri net $\langle E, B, pre, post\rangle$ with the object $\langle\langle E, B, pre\rangle, \langle B, E, post^{op}\rangle\rangle$ of $\mathbf{M}_{\mathbb{N}}\mathbf{Set} \times \mathbf{M}_{\mathbb{N}}\mathbf{Set}^*$.

It follows from Theorem 7.3 that the product of two nets $\langle E_0, B_0, pre_0, post_0\rangle$ and $\langle E_1, B_1, pre_1, post_1\rangle$ in **GNet** is given by the product of $\langle\langle E_0, B_0, pre_0\rangle, \langle B_0, E_0, post_0^{op}\rangle\rangle$ and $\langle\langle E_1, B_1, pre_1\rangle, \langle B_1, E_1, post_1^{op}\rangle\rangle$ in $\mathbf{M}_{\mathbb{N}}\mathbf{Set} \times \mathbf{M}_{\mathbb{N}}\mathbf{Set}^*$.

The product of two objects $\langle E_0, B_0, \alpha_0\rangle$ and $\langle E_1, B_1, \alpha_1\rangle$ in $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$ is the object $\langle E_0 \times E_1, B_0 + B_1, \alpha_0 \wedge \alpha_1\rangle$ where

$$(\alpha_0 \wedge \alpha_1)\langle e_0, e_1, b\rangle = \begin{cases} \alpha_0\langle e_0, b_0\rangle & \text{if } b = (b_0, 0) \\ \alpha_1\langle e_1, b_1\rangle & \text{if } b = (b_1, 1). \end{cases}$$

The details are contained in [DP91]. Clearly, the product of two objects $\langle B_0, E_0, \alpha_0\rangle$ and $\langle B_1, E_1, \alpha_1\rangle$ in $\mathbf{M}_{\mathbb{N}}\mathbf{Set}^*$ can be defined in terms of the product in $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$ as,

$$\begin{aligned} \iota^{-1}(\iota(\langle B_0, E_0, \alpha_0\rangle) \times \iota(\langle B_1, E_1, \alpha_1\rangle)) &= \iota^{-1}(\langle E_0, B_0, \alpha_0^{op}\rangle \times \langle E_1, B_1, \alpha_1^{op}\rangle) \\ &= \iota^{-1}(\langle E_0 \times E_1, B_0 + B_1, \alpha_0^{op} \wedge \alpha_1^{op}\rangle) \\ &= \langle B_0 + B_1, E_0 \times E_1, (\alpha_0^{op} \wedge \alpha_1^{op})^{op}\rangle \\ &= \langle B_0 + B_1, E_0 \times E_1, \alpha_0 \wedge \alpha_1\rangle. \end{aligned}$$

Thus the product of $\langle\langle E_0, B_0, pre_0\rangle, \langle B_0, E_0, post_0^{op}\rangle\rangle$ and $\langle\langle E_1, B_1, pre_1\rangle, \langle B_1, E_1, post_1^{op}\rangle\rangle$ in $\mathbf{M}_{\mathbb{N}}\mathbf{Set} \times \mathbf{M}_{\mathbb{N}}\mathbf{Set}^*$ is given by,

$$\begin{aligned} &\langle\langle E_0, B_0, pre_0\rangle, \langle B_0, E_0, post_0^{op}\rangle\rangle \times \langle\langle E_1, B_1, pre_1\rangle, \langle B_1, E_1, post_1^{op}\rangle\rangle \\ &= \langle\langle E_0, B_0, pre_0\rangle \times \langle E_1, B_1, pre_1\rangle\rangle, \langle B_0, E_0, post_0^{op}\rangle \times \langle B_1, E_1, post_1^{op}\rangle \\ &= \langle\langle E_0 \times E_1, B_0 + B_1, pre_0 \wedge pre_1\rangle\rangle, \langle B_0 + B_1, E_0 \times E_1, post_0^{op} \wedge post_1^{op}\rangle \\ &= \langle\langle E_0 \times E_1, B_0 + B_1, pre_0 \wedge pre_1\rangle\rangle, \langle B_0 + B_1, E_0 \times E_1, (post_0 \wedge post_1)^{op}\rangle \end{aligned}$$

which we identify with the net $\langle E_0 \times E_1, B_0 + B_1, pre_0 \wedge pre_1, post_0 \wedge post_1\rangle$.

Coproducts and the symmetric monoidal closed structure of $\mathbf{M}_{\mathbb{N}}\mathbf{Set}$ lift to **GNet** in an entirely similar fashion.

References

[Amb91] S. J. Ambler. *First Order Linear Logic in Symmetric Monoidal Closed Categories*, PhD thesis, University of Edinburgh, 1991.

[AS79] J. R. Abrial and S. A. Schuman. *Non-deteministic system specification*, in G, Kahn, editor, *Semantics of Concurrent Computation.* Springer-Verlag, LNCS 70, 1979.

[Bar79] M. Barr. $*$-*Autonomous Categories*, LNM, Springer-Verlag, 1979.

[BG] C. T. Brown and D. J. Gurr. *A categorical linear framework for petri nets*, to appear in *Information and Computation.*

[BG90] C. T. Brown and D. J. Gurr. *A categorical linear framework for petri nets*, in *Proc. LICS*, 1990.

[Bro90] C. T. Brown. *Linear Logic and Petri Nets: Categories, Algebra and Proof*, PhD thesis, University of Edinburgh, 1990.

[BG80] R. M. Burstall and J. A. Goguen. *The semantics of clear, a specification language*, in *Proc. of Advance Course on Abstract Software Specifications*, pages 292–332. Springer-Verlag, LNCS 86, 1980.

[DMM89] P. Degano, J. Meseguer, and U. Montanari. *Axiomatising net computations and processes*, in *Proc LICS*, 1989.

[DP88] V. C. V. De Paiva. *The Dialectica Categories*, PhD thesis, University of Cambridge, 1988.

[DP89] V. C. V. De Paiva. *A dialectica-like model of linear logic*, in D.H. Pitt, D.E. Rydeheard, P. Dybjer, A.M. Pitts, and A. Poigné, editors, *Proc. Category Theory and Computes Science, Manchester.* Springer Verlag (LNCS 389), 1989.

[DP91] V. C. V. De Paiva. *A category of multirelations*, Technical report number 255, Univ. of Cambridge, 1991.

[Fou80] M. P. Fourman. *The logic of topoi*, in J. Barwise, editor, the Handbook of Mathematical Logic.

[Gir87] J- Y. Girard. *Linear logic*, TCS, 50:1–102,1987.

[GL87] J- Y. Girard and Y. Lafont. *Linear logic and lazy computation*, in *Proc. TAPSOFT 87 (Pisa)*, volume 2, pages 52–66, 1987. Springer-Verlag, (LNCS 250).

[HDP90] J. M. E. Hyland and V. C. V. De Paiva. *Lineales*, manuscript, October 1990.

[Jen90] K. Jensen *Coloured petri nets: a high level language for system design analysis*, in G. Rosenberg, editor, *Proc. Advances in Petri nets*, 1990. Springer-Verlag, (LNCS).

[Joh77] P. T. Johnstone. *Topos Theory*, Academic Press, 1977.

[Jon86] C. B. Jones. *Systematic Software Development using VDM*, Prentice Hall, 1986.

[Law73] F. W. Lawvere. *Metric spaces, generalised logic, and closed categories*, in *Rend. del Seminario Matematico e Fisico di Milano*, volume 43, 1973.

[LS86] J. Lambek and P. J. Scott. *Introduction to higher order categorical logic*, CUP, 1986.

[LS91] Y. Lafont and T. Streicher. *Games semantics for linear logic*, in *Proc. LICS*, 1991.

[Mil89] R. Milner. *Communication and Concurrency*, Prentice Hall, 1989.

[MM88a] J. Meseguer and U. Montanari. *Petri nets are monoids: A new algebraic foundation for net theory*, in *Proc LICS*, 1988.

[MM88b] J. Meseguer and U. Montanari. *Petri nets are monoids: A new algebraic foundation for net theory*, TechnicaI Report SRICSL-88-3, C.S. Lab., SRI International, January 1988.

[MOM89] N. Marti-Oliet and J. Meseguer. *From petri nets to linear logic*, In D.H. Pitt, D.E. Rydeheard, P. Dybjer, A.M. Pitts, and A. Poigné, editors, *Proc. Category Theory and Computer Science, Manchester*, 1989.

[NRT90] M. Nielsen, G. Rosenberg, and P. S. Thiagarajan. *Elementary transition systems*, Technical Report DAIMI PB – 310, DAIMI, Århus University, April 1990.

[NW91] M. Nielsen and G. Winskel. *Concurrency*, Chapter in the forthcoming *Handbook of Theoretical Computer Science.*

[Rei85] W. Reisig. *Petri Nets: an Introduction*, EATCS Monographs on Theoretical Computer Science Springer-Verlag, 1985.

[See89] R. Seeley. *Linear logic, $*$-autonomous categories and cofree coalgebras*, in *Categories in Computer Science and Logic*, volume 43. AMS, 1989.

[ST88] D. Sannella and A. Tarlecki. *Toward formal development of programs form algebraic specifications: implementations revisited*, Acta Informatica, 25:233–281, 1988.

[Ver91] D. Verity. *Enrichments*, manuscript, February 1991.

[Win84] G. Winskel. *Categories of models for concurrency*, in S.D. Brookes, A.W. Roscoe, and G. Winskel, editors, *Proc. Seminar on Semantics of Concurrency (CMU, Pittsburgh)*, 1984. Springer-Verlag, (LNCS 197).

[Win87] G. Winskel. *Petri nets, algebras, morphisms, and compositionality, Information and Computation*, 72:197–238, 1987.

[Win88] G. Winskel. *A category of labelled petri nets and compositional proof system*, in *Proc LICS*, 1988.

[Wir71] N. Wirth. *Program development by stepwise refinement*, Comm. ACM, 14(4):221–227, 1971.