On weighted Petri net transducers

Robert Lorenz, Markus Huber, Günther Wirsching

Angaben zur Veröffentlichung / Publication details:

Lorenz, Robert, Markus Huber, and Günther Wirsching. 2014. "On weighted Petri net transducers." In *Application and theory of petri nets and concurrency: 35th International Conference, PETRI NETS 2014, Tunis, Tunisia, June 23-27, 2014*, edited by Gianfranco Ciardo and Ekkart Kindler, 8489:233–52. Cham: Springer International. https://doi.org/10.1007/978-3-319-07734-5_13.

On Weighted Petri Net Transducers

Robert Lorenz[1], Markus Huber[1], and Günther Wirsching[2]

[1] Department of Computer Science
University of Augsburg, Germany
firstname.lastname@informatik.uni-augsburg.de
[2] Mathematisch-Geographische Fakultät
Catholic University of Eichstätt, Germany
guenther.wirsching@ku.de

Abstract In this paper we present a basic framework for weighted Petri net transducers (PNTs) for the translation of partial languages (consisting of partial words) as a natural generalisation of finite state transducers (FSTs).

Concerning weights, we use the algebraic structure of continuous concurrent semirings which is based on bisemirings and induces a natural order on its elements. Using the operations of this algebra, it is possible to define the weight of sequential parallel partial words in a standard way. We define the weight of a general partial word as the supremum of the weights of all of its sequential parallel extensions. As a fundamental result we show that concurrent semirings are the least restrictive idempotent bisemiring structure such that partial words with fewer dependencies have bigger weights. Moreover, the weight definition turns out to be compositional, i.e. the weight of (sequential or parallel) composed partial words equals the corresponding bisemiring composition of the weights of its components.

To be able to create complex PNTs through composition of simple PNTs, we introduce clean PNTs and the composition operations union, product, closure, parallel product and language composition on clean PNTs, lifting standard composition operations on FSTs. Composed PNTs yield a compositional computation of weights, where in the case of language composition such a compositional computation is possible only in restricted cases. Moreover, we give definitions for equivalent PNTs and show that all composition operations preserve equivalence. We also show that under certain conditions concerning the algebraic weight structure an FST can be represented by an equivalent PNT.

Keywords: Petri Net, Petri Net Transducer, Weighted Transducer, Labelled Partial Order, Weighted Labelled Partial Order, Partial Language, Semiring, Bisemiring, Concurrent Semiring, Cleanness.

1 Introduction

Weighted finite automata are classical non-deterministic finite automata in which transitions carry weights [6]. These weights may represent cost, time consumption or probability of a transition execution. The behaviour of such automata is defined by a function associating with each word the weight of its execution. For a uniform definition of the behaviour, the set of weights is equipped with the underlying algebraic structure of a

semiring. A semiring provides two operations of binary addition and multiplication of weights. The multiplication is used for determining the weight of a path, and the weight of a word is obtained by the sum of the weights of its underlying paths. If each transition additionally is equipped with an output symbol, the resulting automaton is called a transducer. Transducers are used for the translation between languages over different alphabets for example in natural language processing. For weighted finite automata and transducers (also called finite state transducers or FSTs) there are efficient implementations of composition and optimisation operations in standard libraries [17,27].

There are generalisations to weighted automata over discrete structures other than finite words, some of them introducing concurrency into the model through considering labelled partial orders (LPOs) (also called *partial words* [11] or *pomsets* [18]), not consisting of a total order on their symbols but of a partial order. In [9] an overview is given on weighted finite automata (and transducers) processing tree structures. They are used to recognise weighted context-free languages with weights coming from semirings and do not consider concurrency. In [8] weighted asynchronous cellular automata accepting weighted traces, a special restricted kind of LPOs, are described. Here also only semirings are used to describe weights, i.e. no difference is made between the combination of weights of transitions occurring in sequential order and occurring in parallel. In [14] weighted branching automata accepting weighted sequential parallel LPOs (sp-LPOs), which can be constructed from singletons using operations of sequential and parallel composition, are introduced. Weights now come from bisemirings where the algebraic structure of semirings is extended by a third operation of parallel multiplication (which in this case needs no unit) used for the combination of weights of concurrent transition occurrences. For all these automata models there are widely developed theories concerning equivalent representations as rational expressions or logic formulae, useful composition operations and closure properties [6]. Another extended automata model are Q-Automata [4] whose computations are step sequences. Q-Automata are coined for application in quality management with weights modelling costs and coming from a bisemiring, whose parallel multiplication may not be commutative.

The aim of this paper is the generalisation of automata based weighted transducers through weighted Petri net transducers (PNTs). A PNT is essentially a *place/transition net (PT-net)* having transitions equipped with input symbols, output symbols and weights. An LPO over the set of input symbols is translated into an LPO over the set of output symbols via weighted LPO-runs of the net, where weights are coming from an algebraic bisemiring structure. Thus, PNTs define (in a natural way) the translation between partial languages, consisting of general LPOs instead of words, over different alphabets. In this sense PNTs are a natural generalisation of automata based transducers working on finite words, traces or sp-LPOs. There are already several publications introducing PNTs and applying them in different application areas [24,23,21], however these are mainly case studies. Up to now there is no common basic formal definition and no theoretical development. Moreover, all existing definitions only make use of sequential semantics of PNTs and do not consider weights. In [15,16] we introduced first rather informal definitions of syntax and semantics of PNTs and of composition operations applied to small case studies in the area of semantic dialogue modelling. Another Petri net model with transitions having assigned weights are stochastic Petri

nets (SPNs). SPNs introduce a temporal specification of probabilistic nature and are applied to the performance analysis of timed systems. The weights have no underlying algebraic structure and are used to compute firing probabilities of untimed transitions.

We use a special bisemiring structure called *concurrent semirings* [13][1] to represent weights. Concurrent semirings are a bisemiring structure with some additional laws interrelating its operations. They where already used by Gischer [10], who showed that the set of all extension closed sets of LPOs can be equipped with algebraic operations yielding a concurrent semiring. In particular, concurrent semirings have an idempotent addition inducing a natural order on the set of weights. This feature allows to define the weight of a general LPO in a natural way as the supremum of all weights of its sequential parallel extensions w.r.t. this order. As a fundamental result we show that concurrent semirings are the least restrictive idempotent bisemiring structure such that LPOs with fewer dependencies have bigger weights. Moreover, this weight definition turns out to be compositional, i.e. the weight of (sequential or parallel) composed LPOs equals the corresponding bisemiring composition of the weights of its components.

In practical applications, it is important to be able to create complex transducers through composition of simple ones. To this end we introduce *cleanness* of PNTs and composition operations of union, product, closure, parallel product and language composition on clean PNTs, lifting standard composition operations on FSTs. Cleanness ensures that runs always terminate properly and is shown to be preserved by the above operations. Concerning language composition, we consider different possible adaptions of the FST case and show that concrete constructions yield a compositional computation of weights only in restricted cases.

Since transitions also may have empty input and/or empty output, there are always (infinitely) many PNTs having the same input output behaviour. Such PNTs are equivalent (adapting the notion of equivalent FSTs). We show that the mentioned composition operations preserve equivalence of PNTs. Under certain conditions concerning the algebraic weight structure an FST can be represented by an equivalent PNT.

The presented framework mainly aims at an application in the field of semantic dialogue modelling as described in [26]. In [15,16] we propose the translation between utterances and meanings using PNTs. Since meanings are represented by arbitrary LPOs which need not be sequential parallel, it is not possible to use one of the mentioned weighted automata models. Additionally, PNTs also may be used to model quantitative aspects of computation by adding bisemiring costs to process calculi represented by arbitrary LPOs, generalising the models from [8,14,4].

The paper is organised as follows: In section 2 we recall basic definitions, including LPOs, Petri nets and weighted FSTs. In section 3 we introduce concurrent semirings and weighted LPOs, and examine fundamental relationships between the weight of LPOs and the algebraic weight structure of concurrent semirings. Then we give syntax and semantics of PNTs, define cleanness of PNTs and equivalences on PNTs and examine the representation of FSTs by equivalent PNTs. In section 4 we consider several composition operations of clean PNTs and the preservation of cleanness and equivalence

[1] In [13] concurrent semirings are applied in a trace model of programme semantics. Another axiomatic approach to partial order semantics using algebraic structures extending semirings by an additional operation of concurrent composition is [3] using the notion of trioids.

under these operations. Finally, we give a brief conclusion and outlook on future work in section 5.

All figures in this paper showing PNTs were generated with $\text{PNT}_\varepsilon^{\text{ooL}}$. $\text{PNT}_\varepsilon^{\text{ooL}}$ is a python library for the modular construction of PNTs through composition operations. Constructed PNTs can be exported in all standard picture formats and in an XML-format based on the standard PNML format. $\text{PNT}_\varepsilon^{\text{ooL}}$ will serve as a basis for the implementation and evaluation of algorithms for analysis, simulation and optimisation of PNTs. Its basic functionalities were developed in the bachelor thesis [20].

2 Basic Definitions and Notations

In this section we recall basic definitions and mathematical notations.

2.1 Mathematical Preliminaries

By $\mathbb{N}_0$ we denote the set of *non-negative integers*, by $\mathbb{N}$ the set of *positive integers*. Given a finite set X, the symbol $|X|$ denotes the *cardinality* of X.

The set of all *multisets* over a set X is the set ${\mathbb{N}_0}^X$ of all functions $f : X \to \mathbb{N}_0$. Addition $+$ on multisets is defined by $(m+m')(x) = m(x) + m'(x)$. The relation $\leq$ between multisets is defined through $m \leq m' \iff \exists m''(m+m'' = m')$. We write $x \in m$ if $m(x) > 0$. A set $A \subseteq X$ is identified with the multiset m satisfying $m(x) = 1 \iff x \in A \wedge m(x) = 0 \iff x \notin A$. A multiset m satisfying $m(a) > 0$ for exactly one element a we call *singleton multiset* and denote it by $m(a)a$.

Given a binary relation $R \subseteq X \times Y$ and a binary relation $S \subseteq Y \times Z$ for sets X, Y, Z, their composition is defined by $R \circ S = \{(x,z) \mid \exists y \in Y((x,y) \in R \wedge (y,z) \in S)\} \subseteq X \times Z$. For $X' \subseteq X$ and $Y' \subseteq Y$ the restriction of R onto $X' \times Y'$ is denoted by $R|_{X' \times Y'}$. For a binary relation $R \subseteq X \times X$ over a set X, we denote $R^1 = R$ and $R^n = R \circ R^{n-1}$ for $n \geq 2$. The symbol R^+ denotes the *transitive closure* $\bigcup_{n \in \mathbb{N}} R^n$ of R.

Let A be a finite set of symbols. A *(linear) word* over A is a finite sequence of symbols from A. For a word w its length $|w|$ is defined as the number of its symbols. The symbol ε denotes the *empty word* satisfying $|\varepsilon| = 0$. The empty word is the neutral w.r.t. concatenation of words: $w\varepsilon = \varepsilon w = w$. By A^* we denote the set of all words over A, including the empty word. A *language over A* is a (possibly infinite) subset of A^*. A *step over A* is a multiset over A. A *step sequence* or *step-wise linear word* over A is an element of $(\mathbb{N}_0^A)^*$ and a *step language over A* is a (possibly infinite) subset of $(\mathbb{N}_0^A)^*$.

A *directed graph* is a pair $G = (V, \to)$, where V is a finite *set of nodes* and $\to \subseteq V \times V$ is a binary relation over V, called the *set of edges*. The *preset* of a node $v \in V$ is the set ${}^\bullet v = \{u \mid u \to v\}$. The *postset* of a node $v \in V$ is the set $v^\bullet = \{u \mid v \to u\}$. A *path* is a sequence of (not necessarily distinct) nodes $v_1 \ldots v_n$ $(n > 1)$ such that $v_i \to v_{i+1}$ for $i = 1, \ldots, n-1$. A path $v_1 \ldots v_n$ is a *cycle* if $v_1 = v_n$. A directed graph is called *acyclic* if it has no cycles. An acyclic directed graph $(V, \to')$ is an *extension* of an acyclic directed graph $(V, \to)$ if $\to \subseteq \to'$. An acyclic directed graph $(V', \to)$ is a *prefix* of an acyclic directed graph $(V, \to)$ if $V' \subseteq V$ and $(v' \in V') \wedge (v \to v') \Rightarrow (v \in V')$.

An *irreflexive partial order* over a set V is a binary relation $< \subseteq V \times V$ which is irreflexive $(\forall v \in V : v \not< v)$ and transitive $(<= <^+)$. We identify a finite irreflexive partial

order $<$ over V with the directed graph $(V,<)$. Two nodes $v,v' \in V$ of a irreflexive partial order $po = (V,<)$ are called *independent* if $v \not< v'$ and $v' \not< v$. By $co_< \subseteq V \times V$ we denote the set of all pairs of independent nodes of V. A *reflexive partial order* over V is a binary relation $\leq \subseteq V \times V$ which is reflexive ($\forall v \in V : v \leq v$), transitive and antisymmetric ($\forall v \in V : v \leq w \wedge w \leq v \implies v = w$).

2.2 Labelled Partial Orders

We use irreflexive partial orders labelled by action names to represent single non-sequential runs of concurrent systems. The nodes of such a labelled partial order represent events and its arrows an 'earlier than'-relation between them in the sense that one event can be observed earlier than another event. If there are no arrows between two events, then these events are independent and are called *concurrent*. Concurrent events can be observed in arbitrary sequential order and simultaneously.

Formally, a *labelled partial order* (*LPO*) *over a set* X is a 3-tuple $(V,<,l)$, where $(V,<)$ is a irreflexive partial order and $l : V \to X$ is a labelling function on V. LPOs over X are also called *partial words over* X. In most cases, we only consider LPOs up to isomorphism, i.e. only the labelling of events is of interest, but not the event names. Formally, two LPOs $(V,<,l)$ and $(V',<',l')$ are *isomorphic* if there is a bijective renaming function $I : V \to V'$ satisfying $l(v) = l'(I(v))$ and $v < w \Leftrightarrow I(v) <' I(w)$. If an LPO *lpo* is of the form $(\{v\},\emptyset,l)$, then it is called a *singleton LPO* and denoted by $lpo = l(v)$. We call a set of pairwise non-isomorphic LPOs a *partial language*. If L is a partial language, then an LPO $lpo \in L$ is called *minimal (in L)* if there is no extension of *lpo* in L. In figures, in general we do not show the names of the nodes of an LPO, but only their labels and we often omit transitive arrows of LPOs for a clearer presentation.

A *step-wise linear LPO* is an LPO $(V,<,l)$ where the relation $co_<$ is transitive. The maximal sets of independent events are called *steps*. The steps of a step-wise linear LPOs are linearly ordered. Thus, step-wise linear LPOs can be identified with step sequences. A *step-linearisation* of an LPO *lpo* is a step-wise linear LPO which is an extension of *lpo*. The set of *sequential parallel* LPOs (sp-LPOs) is the smallest set of LPOs containing all singleton LPOs (over a set X) and being closed under the sequential and parallel product of LPOs. The *sequential product* of two LPOs $lpo_1 = (V_1,<_1,l_1)$ and $lpo_2 = (V_2,<_2,l_2)$ is defined by $lpo_1 ; lpo_2 = (V_1 \cup V_2, <_1 \cup <_2 \cup V_1 \times V_2, l_1 \cup l_2)$, where V_1 and V_2 are assumed to be disjoint. Their *parallel product* is defined by $lpo_1 \parallel lpo_2 = (V_1 \cup V_2, <_1 \cup <_2, l_1 \cup l_2)$, where again V_1 and V_2 are assumed to be disjoint. For an LPO *lpo* we denote by $SP(lpo)$ the set of all sequential parallel extensions of *lpo* and by $SP_{min}(lpo)$ the set of all minimal sequential parallel extensions of *lpo* in $SP(lpo)$. If *lpo* is an extension of lpo', we write $lpo \leq lpo'$.

The sequential and parallel product of LPOs is extended to sets of LPOs A,B in the obvious way: $A \parallel B = \{a \parallel b \mid a \in A, b \in B\}$ and $A ; B = \{a ; b \mid a \in A, b \in B\}$. Moreover, we define the closure of a set of LPOs A by $A^* = \{a_1 ; \ldots ; a_n \mid n \in \mathbb{N}, a_i \in A\} \cup \{\varepsilon\}$, where ε denotes the empty LPO.

2.3 Petri Nets

A *net* is a 3-tuple $N = (P,T,F)$, where P is a finite set of *places*, T is a finite set of *transitions* disjoint from P and $F \subseteq (P \times T) \cup (T \times P)$ is the *flow relation*. A *marking* of a net assigns to each place $p \in P$ a number $m(p) \in \mathbb{N}_0$, i.e. a marking is a multiset over P. A *marked net* is a net $N = (P,T,F)$ together with an *initial marking* m_0.

A *place/transition Petri net* (*PT-net*) is a 4-tuple $N = (P,T,F,W)$, where (P,T,F) is a net and $W : (P \times T) \cup (T \times P) \to \mathbb{N}_0$ is a *flow weight function* satisfying $W(x,y) > 0 \Leftrightarrow (x,y) \in F$. For (transition) steps τ over T we introduce the two multisets of places ${}^\bullet\tau(p) = \sum_{t \in T} \tau(t)W(p,t)$ and $\tau^\bullet(p) = \sum_{t \in T} \tau(t)W(t,p)$. A transition step τ *can occur* in m if $m \geq {}^\bullet\tau$. If a transition step τ occurs in m, then the resulting marking m' is defined by $m' = m - {}^\bullet\tau + \tau^\bullet$. We write $m \xrightarrow{\tau} m'$ to denote that τ can occur in m and that its occurrence leads to m'. A *step execution in m* of a PT-net is a finite sequence of multisets of transitions $\sigma = \tau_1 \ldots \tau_n$ such that there are markings $m_1, \ldots, m_n$ satisfying $m \xrightarrow{\tau_1} m_1 \xrightarrow{\tau_2} \ldots \xrightarrow{\tau_n} m_n$. The markings which can be reached from the initial marking m_0 via step executions are called *reachable*.

We use LPOs over T to represent single non-sequential runs of PT-nets, i.e. the events of an LPO represent transition occurrences. An LPO $lpo = (V, <, l)$ over T is an *LPO-run* of a marked PT-net $N = (P,T,F,W,m_0)$ if each step-linearisation of lpo is a step execution of N in m_0. If an LPO-run $lpo = (V, <, l)$ occurs in a marking m, the resulting marking m' is defined by $m' = m - \Sigma_{v \in V} {}^\bullet l(v) + \Sigma_{v \in V} l(v)^\bullet$. We write $m \xrightarrow{lpo} m'$ to denote the occurrence of an LPO-run lpo.

2.4 Weighted Finite State Transducers

Finite-state transducers (FSTs) are finite automata in which each transition is augmented with an output label in addition to the familiar input label. Output labels are concatenated along a path to form an output sequence. Weighted transducers are finite-state transducers in which each transition additionally carries some weight. The weights are elements of an algebraic structure called semiring.

A *semiring* is a quintuple $\mathscr{S} = (S, \oplus, \otimes, \overline{0}, \overline{1})$, where $(S, \oplus, \overline{0})$ is a commutative monoid, $(S, \otimes, \overline{1})$ is a monoid, $\otimes$ (the *S-multiplication*) distributes over $\oplus$ (the *S-addition*) from both sides of $\otimes$ and the zero $\overline{0}$ is absorbing w.r.t. $\otimes$ ($\overline{0} \otimes x = x \otimes \overline{0} = \overline{0}$). If $\otimes$ is commutative, then the semiring is called *commutative*.

The $\otimes$-operation is used to compute the weight of a path of an FST by multiplying the weights of the transitions along that path. The $\oplus$-operation is used to compute the weight of a pair of input and output sequences (u,v) by summing up the weights of all paths labelled with (u,v). A *weighted finite state transducer (FST)* over a semiring $\mathscr{S}$ is an 8-tuple $T = (\Sigma, \Delta, Q, I, F, E, \lambda, \rho)$, where Σ is a finite alphabet of input symbols, Δ is a finite alphabet of output symbols, Q is a finite set of states, $I \subseteq Q$ is the set of initial states, $F \subseteq Q$ is the set of final states, $E \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Delta \cup \{\varepsilon\}) \times S \times Q$ is the finite set of transitions, $\lambda : I \to S$ is the initial weight function and $\rho : F \to S$ is the final weight function. For a transition $e = (q_1, x, y, s, q_2)$, we denote $p(e) = q_1$ to be its *start state*, $n(e) = q_2$ its *next state*, $\omega(e) = s$ its *weight*, $\sigma(e) = x$ its *input label* and $\delta(e) = y$ its *output label*. Two transitions e_1 and e_2 are *consecutive* if $n(e_1) = p(e_2)$. A

sequence $\pi = e_1 \dots e_k \in E^*$ of consecutive transitions is called a *path* with start state $p(\pi) = p(e_1)$ and next state $n(\pi) = n(e_k)$. The input label of a path $e_1 \dots e_k$ is the word $\sigma(e_1) \dots \sigma(e_k)$. The output label of a path $e_1 \dots e_k$ is the word $\delta(e_1) \dots \delta(e_k)$. For subsets $Q_1, Q_2 \subseteq Q$, $u \in \Sigma^*$ and $v \in \Delta^*$, we denote by $P(Q_1, Q_2)$ the set of all paths from states in Q_1 to states in Q_2, $P(Q_1, u, Q_2)$ the subset of all paths from $P(Q_1, Q_2)$ with input label u, $P(Q_1, u, v, Q_2)$ the subset of all paths from $P(Q_1, u, Q_2)$ with output label v. The weight of a path is defined by $\omega(e_1 \dots e_k) = \omega(e_1) \otimes \dots \otimes \omega(e_k)$. The *output weight* of a pair of words $(u, v) \in \Sigma^* \times \Delta^*$ is defined by

$$T(u,v) = \bigoplus_{\pi \in P(I,u,v,F)} \lambda(p(\pi)) \otimes \omega(\pi) \otimes \rho(n(\pi)),$$

when the sum is well-defined in S. This is the case, for example, if the considered semiring is *complete* (see subsection 3.1). If $P(I, u, v, F) = \emptyset$, we set $T(u, v) = \overline{0}$. For a detailed overview on weighted FSTs see for example [17].

3 Definition of Weighted Petri Net Transducers

In this section we introduce weighted Petri net transducers (PNTs) for the translation between partial languages. For taking weights into account, we consider weighted LPOs (WLPOs) which are LPOs with additional node weights. Then the total weight of a WLPO is computed from the node weights using binary operations on the set of weights. We shall infer from a result of Gischer [10] that the algebraic structure of the set of possible weights is not arbitrary: If we postulate that the binary operations on the weights reflect sequential and parallel product of LPOs, then the set of possible weights must admit the algebraic structure *concurrent semiring* [13]. For the translation of input words into output words we equip transitions with input symbols, output symbols and weights and consider weighted LPO-runs. Based on this idea, we define syntax and semantics of PNTs. Then, we define equivalence of PNTs and examine the connection between PNTs and FSTs.

3.1 Continuous Concurrent Semirings

A binary operation $\oplus$ on a set S defines a binary relation on S via $a \leq_\oplus b :\Leftrightarrow a \oplus b = b$. If $\oplus$ is idempotent, associative, and commutative, then this relation is reflexive, transitive, and antisymmetric, hence a reflexive partial order. Moreover, if S is equipped with the partial order $\leq_\oplus$, then $\forall a, b \in S:\ a \oplus b = \sup\{a, b\}$, where the supremum is taken w.r.t. $\leq_\oplus$. If $(S, \oplus, \overline{0})$ is a monoid, and if $T \subseteq S$ is an arbitrary subset, then $\bigoplus T := \bigoplus_{t \in T} t := \sup(T)$, where the supremum of the empty set is understood to be the neutral element of the monoid. A semiring $(S, \oplus, \otimes, \overline{0}, \overline{1})$ is called *idempotent* if $\oplus$ is idempotent. An idempotent semiring is called *continuous* [5] if, for any subset $T \subseteq S$, the supremum is well-defined in S (that means the semiring is *complete*), and $\otimes$ distributes over the supremum from both sides: $\forall s \in S: s \otimes \bigoplus T = \bigoplus_{t \in T} s \otimes t$ and $(\bigoplus T) \otimes s = \bigoplus_{t \in T} t \otimes s$.

A *bisemiring* is a six-tuple $\mathscr{S} = (S, \oplus, \otimes, \boxtimes, \overline{0}, \overline{1})$, where $(S, \oplus, \otimes, \overline{0}, \overline{1})$ is a semiring and $(S, \oplus, \boxtimes, \overline{0}, \overline{1})$ is a commutative semiring.[2] The binary operation $\boxtimes$ on the set S is called *S-parallel multiplication*. If $\otimes$ distributes over $\boxtimes$ from both sides, the bisemiring is called *distributive*, if $\oplus$ is idempotent, the bisemiring is called *idempotent*, and if both semirings $(S, \oplus, \otimes, \overline{0}, \overline{1})$ and $(S, \oplus, \boxtimes, \overline{0}, \overline{1})$ are continuous, the bisemiring is called *continuous*. According to [13], a *concurrent semiring* is an idempotent bisemiring $(S, \oplus, \otimes, \boxtimes, \overline{0}, \overline{1})$ satisfying

$$\forall a, b, c, d \in S: \quad (a \boxtimes b) \otimes (c \boxtimes d) \leq_{\oplus} (a \otimes c) \boxtimes (b \otimes d). \tag{CS}$$

Concurrent semirings will be used to define the weight of a run of a Petri net transducer. $\otimes$ will be used to model the composition of weights of a sequence of runs (as in the case of FSTs) and $\boxtimes$ models the composition of weights of concurrent runs. Therefore, $\boxtimes$ is required to be commutative. The unit $\overline{1}$ can be thought of as the weight of the empty run (the analogue of the empty word). It is shared by $\otimes$ and $\boxtimes$, since the sequential or concurrent execution of a run r and the empty run does not change r. Using $\otimes$ and $\boxtimes$, the weight of a sequential parallel run can be defined in the standard way.

Idempotence of $\oplus$ induces a natural order on the set of weights. We will define the weight of a general run in a natural way as the supremum of all weights of its sequential parallel extensions w.r.t. this order. Condition (**CS**) will ensure that runs with fewer dependencies have bigger weights.

Example 1. If $\mathscr{S} = (S, \oplus, \otimes, \overline{0}, \overline{1})$ is an idempotent semiring such that $\leq_{\oplus}$ is a total order and $\overline{1}$ is maximal w.r.t. to that order, then we have $\mathscr{S} = (S, \max, \otimes, \overline{0}, \overline{1})$, and $(S, \max, \otimes, \min, \overline{0}, \overline{1})$ is a concurrent semiring extending $\mathscr{S}$.

If $\mathscr{S} = (S, \oplus, \otimes, \overline{0}, \overline{1})$ is an idempotent and commutative semiring, then the *doubled semiring* $(S, \oplus, \otimes, \otimes, \overline{0}, \overline{1})$ is a concurrent semiring extending $\mathscr{S}$.

Example 2. Based on the well-known *Viterbi semiring* $([0,1], \max, \cdot, 0, 1)$ representing probabilities of actions, the structure $\mathscr{V} := ([0,1], \max, \cdot, \min, 0, 1)$ yields a continuous concurrent semiring.

The structure $\mathscr{T} := ([0,\infty], \min, +, \max, \infty, 0)$ is a continuous concurrent semiring. It is based on the well-known *tropical semiring* $([0,\infty], \min, +, \infty, 0)$ representing execution times of actions.

Note that $\mathscr{V}$ and $\mathscr{T}$ are isomorphic, e.g. an isomorphism is given by $t = -\log(v)$. Both concurrent semirings extend a semiring as in the first construction of example 1.

An example of a concurrent semiring, which is not of the above kind, is $\mathscr{A} := (\{-\infty\} \cup [0,\infty[, \max, +, \boxtimes, -\infty, 0)$, where $a \boxtimes b := a + b + \min(a,b)$. It is based on the *arctic semiring*.

3.2 Weighted LPOs

We use LPOs extended by weights from a bisemiring to model runs of PNTs. By definition, a *weighted LPO* (WLPO) over a alphabet $\mathscr{A}$ and a bisemiring $\mathscr{S} = (S, \oplus, \otimes, \boxtimes,$

[2] In particular, both multiplications share the same unit. A similar algebraic structure without requiring commutativity of the second semiring is defined in [4], where it is called *Q-Algebra* and coined for application in quality management. In [14] a slightly different notion of bisemirings is used where parallel multiplication may miss a unit.

$\overline{0},\overline{1})$ is a quadruple $(V,<,l,\nu)$ such that $(V,<,l)$ is an LPO over $\mathscr{A}$ and $\nu : V \to S$ is an additional *weight function*. We use all notions introduced for LPOs also for WLPOs.

The weight of sp-WLPOs can be defined similar as in [14] for runs of so called weighted branching automata. The total weight of an sp-WLPO is computed from the weights of their nodes through applying $\otimes$ to the sequential product and $\boxtimes$ to the parallel product of sub-WLPOs.

Definition 1 (Weight of sp-WLPOs). *The weight $\omega(wlpo)$ of an sp-WLPO $wlpo = (V,<,l,\nu)$ over a bisemiring is defined inductively as follows:*

- *If $V = \{v\}$, then $\omega(wlpo) = \nu(v)$.*
- *If $wlpo = wlpo_1 ; wlpo_2$, then $\omega(wlpo) = \omega(wlpo_1) \otimes \omega(wlpo_2)$.*
- *If $wlpo = wlpo_1 \parallel wlpo_2$, then $\omega(wlpo) = \omega(wlpo_1) \boxtimes \omega(wlpo_2)$.*

This is the standard technique to define weights of sequential parallel LPOs [14] with weights coming from a bisemiring. In particular, the given weight of sp-WLPOs is well-defined, since the set of sp-WLPOs as well as the sub-structure $(S,\otimes,\boxtimes)$ of a bisemiring $(S,\oplus,\otimes,\boxtimes,\overline{0},\overline{1})$ form an sp-algebra admitting an sp-algebra homomorphism from the set of sp-WLPOs into the bisemiring. Note that for WLPOs $wlpo = (V,<,l,\nu)$ with underlying total order $V = \{v_1 < \cdots < v_n\}$ the weights computes $\omega(wlpo) = \bigotimes_{i=1}^{n} \nu(v_i)$, i.e. the above definition is compatible with the weight definition of paths of an FST. We now propose a weight definition for general WLPOs.

Definition 2 (Sequential-Parallel Weight of WLPOs). *Let $wlpo = (V,<,l,\omega)$ be a WLPO. Then its* sp-weight *is defined by $\omega_{sp}(wlpo) = \bigoplus_{wlpo' \in SP(wlpo)} \omega(wlpo')$.*

If the bisemiring of weights is idempotent, then the sp-weight of a WLPO-run equals the maximal weight of its sequential parallel extensions. As a fundamental result we will show that condition (**CS**) of concurrent semirings are the minimal requirement on idempotent bisemirings such that less restrictive weighted LPOs yield bigger weights. Moreover, the use of concurrent semirings ensures that the sp-weight of WLPOs can be computed in a modular way using bisemiring-operations.

Theorem 1. *Let $\mathscr{A}$ be an alphabet and $\mathscr{S} = (S,\oplus,\otimes,\boxtimes,\overline{0},\overline{1})$ an idempotent bisemiring. Then the following assertions are equivalent:*

(A) *If u_1,u_2 are sp-WLPOs over $\mathscr{A}$ and $\mathscr{S}$ and if u_1 is an extension of u_2, then $\omega(u_1) \leq_{\oplus} \omega(u_2)$.*
(B) *$\mathscr{S}$ is a concurrent semiring.*

Proof. (A) $\Rightarrow$ (B): Let $\mathscr{I}$ be the set of sets of sp-LPOs over $\mathscr{A}$ which are *ideals* as introduced by Gischer [10] as extension-closed sets of sp-LPOs. As proved by Gischer, $(\mathscr{I},\cup,;,\|_I,\emptyset,\{\varepsilon\})$ is a concurrent semiring, where $A \|_I B$ is defined as the least ideal containing $A \parallel B$ and ε is the empty LPO. We show that the mapping $\omega : \mathscr{I} \to S$ defined by $\omega(A) = \bigoplus_{a \in A} \omega(a)$ is a bisemiring-homomorphism. Let $A,B \in \mathscr{I}$, then

- $\omega(A \cup B) = \omega(A) \oplus \omega(B)$, since $\oplus$ is assumed to be idempotent, commutative and associative.

- $\omega(A;B) = \omega(A) \otimes \omega(B)$ follows from $\omega(a;b) = \omega(a) \otimes \omega(b)$ for $a \in A, b \in B$ and because $\otimes$ distributes over $\oplus$.
- We claim that $\omega(A \parallel_I B) = \bigoplus_{a \in A, b \in B} \omega(a \parallel b) = \omega(A) \boxtimes \omega(B)$. The second equation follows from $\omega(a \parallel b) = \omega(a) \boxtimes \omega(b)$ for $a \in A, b \in B$ and because $\boxtimes$ distributes over $\oplus$. The first equation follows from (A), since each LPO in $A \parallel_I B$ is an extension of an LPO of the form $a \parallel b$ with $a \in A, b \in B$.

This proves that $\mathscr{S}$ also is a concurrent semiring.

(B) $\Rightarrow$ (A): Let u_1 be a proper extension of u_2 such that u_1, u_2 are non-isomorphic. Gischer shows in [10] in the Interpolation Lemma that the least ideal containing u_1 can be transformed in finitely many steps into the least ideal containing u_2 using one of the concurrent semiring equations w.r.t. the operations of $(\mathscr{I}, \cup, ;, \parallel_I, \emptyset, \{\varepsilon\})$ in each step, where at least once the equation $(a \parallel_I b);(c \parallel_I d) <_\cup (a;c) \parallel_I (b;d)$ (which corresponds to condition (**CS**) of concurrent semirings) is applied. Denote $u_i = (V_i, <_i, l_i)$ and equip the nodes of each u_i with the weight function ω yielding WLPOs. Then both $u_i' = (V_i, <_i, \omega)$ are sp-LPOs over S. Since $\mathscr{S}$ is a concurrent semiring, it is now possible to transform u_1' into u_2' using the same sequence of concurrent semiring equations as for the transformation of u_1 into u_2, but now w.r.t. the concurrent semiring operations of $\mathscr{S}$. We deduce $\omega(u_1) = u_1' \leq_\oplus u_2' = \omega(u_2)$. □

We deduce that the sp-weight can be computed in a modular way.

Lemma 1. *The following holds for LPOs lpo_1 and lpo_2:*

(i) $SP(lpo_1;lpo_2) = \{lpo_1';lpo_2' \mid lpo_1' \in SP(lpo_1), lpo_2' \in SP(lpo_2)\}$,
(ii) $SP_{min}(lpo_1 \parallel lpo_2) = \{lpo_1' \parallel lpo_2' \mid lpo_1' \in SP_{min}(lpo_1), lpo_2' \in SP_{min}(lpo_2)\}$.

Proof. Straightforward observation. □

Theorem 2. *Let $\mathscr{A}$ be an alphabet and $\mathscr{S} = (S, \oplus, \otimes, \boxtimes, \overline{0}, \overline{1})$ a concurrent semiring. Then the following assertions hold for weighted LPOs $wlpo_1, wlpo_2$ over $\mathscr{A}$ and $\mathscr{S}$:*

(C) $\omega_{sp}(wlpo_1;wlpo_2) = \omega_{sp}(wlpo_1) \otimes \omega_{sp}(wlpo_2)$.
(D) $\omega_{sp}(wlpo_1 \parallel wlpo_2) = \omega_{sp}(wlpo_1) \boxtimes \omega_{sp}(wlpo_2)$.

Proof. ad (C): We apply distributivity of $\otimes$ over $\oplus$ in the formula given in definition 2, and use the set equation (i) of the previous lemma.

ad (D): We claim $\omega_{sp}(wlpo_1 \parallel wlpo_2) = \bigoplus_{wlpo_i' \in SP_{min}(wlpo_i)} \omega(wlpo_1') \boxtimes \omega(wlpo_2')$. This equation follows from (A) in the previous theorem using the set equation (ii) of the previous lemma in the formula given in definition 2. The statement follows now from distributivity of $\boxtimes$ over $\oplus$. □

Example 3. Consider the concurrent semiring $\mathscr{V}$ defined in subsection 3.1. The decision for min as parallel multiplication can be interpreted as follows: If $wlpo = wlpo_1 \parallel wlpo_2$, then $wlpo_1$ and $wlpo_2$ are both necessary but independent parts of the run $wlpo$ of a concurrent system and the probability of $wlpo$ cannot be better than the probability of one of its parts. In [25] we give a justification for that choice of min in the context of semantic dialogue modelling.

Consider the concurrent semiring $\mathscr{T}$ defined in subsection 3.1. It can be used to compute the minimal execution time of a run given by an arbitrary WLPO $wlpo$ of a concurrent system.

3.3 Syntax of Petri Net Transducers

A PNT is a Petri net which, for every transition occurrence, may read a symbol x from an input alphabet Σ and may print a symbol y from an output alphabet Δ. Additionally, a weight s from a bisemiring is assigned to each transition. If no input symbol should be read or no output symbol should be printed, we use the empty word symbol ε as annotation. We use the basic Petri net class of PT-nets to define PNTs.

Definition 3 (Petri Net Transducer). *A* Petri net transducer (PNT) *over a bisemiring* $\mathscr{S} = (S, \oplus, \otimes, \boxtimes, \overline{0}, \overline{1})$ *is a tuple* $N = (P, T, F, W, p_I, p_F, \Sigma, \sigma, \Delta, \delta, \omega)$*, where*

- (P, T, F, W) *is a marked PT-net (called the* underlying PT-net*),* $p_I \in P$ *is the* source place *satisfying* ${}^\bullet p_I = \emptyset$ *and* $p_F \in P$ *is the* sink place *satisfying* $p_F^\bullet = \emptyset$*,*
- Σ *is a set of* input symbols *and* $\sigma : T \to \Sigma \cup \{\varepsilon\}$ *is the* input mapping*,*
- Δ *is a set of* output symbols *and* $\delta : T \to \Delta \cup \{\varepsilon\}$ *is the* output mapping*.*
- $\omega : T \to S$ *is the* weight function*.*

We call the marking $m_0 = p_I$ *the* initial marking *and* $m_F = p_F$ *the* final marking*. A PNT is called* clean *if the final marking is the only reachable marking* m *with* $m(p_F) > 0$*.*

A WLPO $wlpo = (V, <, l, \nu)$ *over* T *is a* weighted LPO-run *of* N *if the underlying LPO* $lpo = (V, <, l)$ *is an LPO-run of* N *with* $m_0 \xrightarrow{lpo} m_F$ *and if* $\nu(v) = \omega(l(v))$*. We denote by* $WLPO(N)$ *the set of all weighted LPO-runs of* N*.*

The cleanness property is similar to cleanness of Boxes [2] or soundness of workflow nets [22] and ensures that PNT semantics are closed under (sequential) product and closure. The final marking can be reached only from a finite set of reachable markings [12]. There are some obvious differences to the syntax of FSTs. There is only one initial state instead of multiple initial states, one final state instead of multiple final states and there are no initial and final weight functions. It is obvious that these restrictions are no real limitation. Figure 1 shows examples of PNTs, where an input symbol x, output symbol y and weight s are annotated to a transition in the form $x{:}y/s$, and annotations of the form $\varepsilon{:}\varepsilon/1$ are not shown.

3.4 Semantics of Petri Net Transducers

Considering non-sequential semantics of Petri nets, a PNT can be used to translate a partial language into another partial language, where so called input words are related to so called output words. Input and output words are defined as LPOs $(V, <, l)$ with a labelling function $l : V \to \mathscr{A} \cup \{\varepsilon\}$ for some input or output alphabet $\mathscr{A}$. Such LPOs we call ε*-LPOs*. For each ε-LPO $(V, <, l)$ we construct the *corresponding* ε*-free LPO* $(W, <|_{W \times W}, l|_W)$, $W = V \setminus l^{-1}(\varepsilon)$ by deleting ε-labelled nodes together with their adjacent edges. Since partial orders are transitive, this does not change the order between the remaining nodes.

Definition 4 (Input and Output Labels of Runs). *Let* $N = (P, T, F, W, p_I, p_F, \Sigma, \sigma, \Delta, \delta, \omega)$ *be a PNT and let* $wlpo = (V, <, l, \nu) \in WLPO(N)$*. The* input label *of* $wlpo$ *is the LPO* $\sigma(wlpo)$ *corresponding to the* ε*-LPO* $(V, <, \sigma \circ l)$*. The* output label *of* $wlpo$ *is the LPO* $\delta(wlpo)$ *corresponding to the* ε*-LPO* $(V, <, \delta \circ l)$*.*

For LPOs u over Σ and v over Δ, we denote by $WLPO(N,u)$ the subset of all WLPOs wlpo from $WLPO(N)$ with input label $\sigma(wlpo) = u$, and by $WLPO(N,u,v)$ the subset of all WLPOs from $WLPO(N,u)$ with output label $\delta(wlpo) = v$.

The input language $L_I(N)$ *of N is the set of all input labels of weighted LPO-runs. Its elements are called* input words. *The* output language $L_O(N)$ *of N is the set of all output labels of weighted LPO-runs. Its elements are called* output words.

The language $L(N)$ *of N is the set of all pairs of LPOs (u,v) over $\Sigma \times \Delta$ with $WLPO(N,u,v) \neq \emptyset$.*

Note that the input and output language of a PNT N are extension closed, since $WLPO(N)$ is extension closed. The *output weight* of a PNT assigned to all pairs of LPOs u over Σ and v over Δ is based on weights of its WLPO-runs.

Definition 5 (Output Weight of PNTs). *Let $N = (P,T,F,W,p_I,p_F,\Sigma,\sigma,\Delta,\delta,\omega)$ be a PNT over a concurrent semiring $\mathscr{S} = (S,\oplus,\otimes,\boxtimes,\overline{0},\overline{1})$, u be an LPO over Σ and v be an LPO over Δ. The* output weight $N(u,v)$ *is defined by*

$$N(u,v) = \bigoplus_{wlpo \in WLPO(N,u,v)} \omega_{sp}(wlpo),$$

when this sum is well-defined in S (note that the sum may be infinite). We set $N(u,v) = \overline{0}$ if $WLPO(N,u,v) = \emptyset$.

Note that the output weight equals the supremum of all weights of corresponding runs, since $\oplus$ is idempotent. If the concurrent semiring is continuous, the supremum always exists in S [5]. From the considerations in subsection 3.2 we immediately deduce that it is enough to consider minimal weighted sp-runs in the defining sum of the output weight using condition (**CS**) of concurrent semirings.

Corollary 1. *Let $N = (P,T,F,W,p_I,p_F,\Sigma,\sigma,\Delta,\delta,\omega)$ be a PNT over a concurrent semiring $\mathscr{S} = (S,\oplus,\otimes,\boxtimes,\overline{0},\overline{1})$, u be an LPO over Σ and v be an LPO over Δ. Then $N(u,v) = \bigoplus_{wlpo \in WLPO_{min}(N,u,v), wlpo' \in SP_{min}(wlpo)} \omega(wlpo')$, when this sum is well-defined in S, where $WLPO_{min}(\cdot)$ is the subset of all minimal WLPOs in $WLPO(\cdot)$.*

3.5 Equivalent PNTs

Concerning PNT semantics, only the input output behaviour is relevant. Since transitions also may have empty input and/or empty output, there are always (infinitely) many PNTs having the same semantics. For practical application, such PNTs are equivalent. We introduce equivalent PNTs lifting the corresponding notion for FSTs.

Definition 6 (Equivalent PNTs). *Let N_1, N_2 be two PNTs.*

(a) N_1 and N_2 are called structure equivalent *if $L(N_1) = L(N_2)$.*
(b) If N_1 and N_2 are structure equivalent, then they are called output equivalent *if $N_1(u,v) = N_2(u,v)$ for all $(u,v) \in L(N_1) = L(N_2)$.*

Two structure equivalent PNTs perform the same translation between input and output words, but the weights of these translations may be different. Two output equivalent PNTs perform the same weighted translation between input and output words, but the distribution of weights within WLPO-runs may be different. Output equivalence is usually used in the context of FSTs in order to push weights along paths [17]. Weight pushing leads to output equivalent FSTs which allow for more efficient FST-algorithms.

Example 4. In the following, consider a fixed concurrent semiring serving as the set of weights. We denote by $N(a,b,w)$ the clean PNT consisting of no other places than the source and sink place and exactly one transition with input symbol a, output symbol b and weight w connecting the source with the sink place. Then the following PNTs are structure equivalent: $N_1 = N(a,b,w)$, $N_2 = N(a,\varepsilon,u) \otimes N(\varepsilon,b,v)$ and $N_3 = N(a,\varepsilon,x) \boxtimes N(\varepsilon,b,y)$. They are output equivalent if $w = u \otimes v = x \boxtimes y$. The following PNTs in general also are output equivalent: $N_2 = N(a,\varepsilon,u) \otimes N(\varepsilon,b,v)$, $N_5 = N(a,\varepsilon,v) \otimes N(\varepsilon,b,u)$ and $N_6 = N(a,\varepsilon,u \otimes v) \otimes N(\varepsilon,b,\overline{1})$.

Since not each semiring can be extended to a concurrent semiring, not each FST can be represented by an equivalent PNT. On the other side, each finite automaton can be represented by a special PT-net, a so called *state machine*, having the same set of runs. This means, if a semiring can be extended to a concurrent semiring, then an FST over this semiring is output equivalent to a PNT. Given an FST over a semiring satisfying the preconditions in one of the cases considered in example 1, the constructions given in the example may be used to define an equivalent PNT:

Proposition 1. *Let FST $T = (\Sigma,\Delta,Q,I,F,E,\lambda,\rho)$ be a weighted FST over a semiring $\mathscr{S} = (S,\oplus,\otimes,\overline{0},\overline{1})$ satisfying one of the following conditions: (1) $\mathscr{S}$ is idempotent and commutative, (2) $\mathscr{S}$ is idempotent such that $\leq_\oplus$ is a total order and $\overline{1}$ is maximal w.r.t. to that order. Then there is a PNT $N = (P,T,F,W,p_I,p_F,\Sigma,\sigma,\Delta,\delta,\omega)$ satisfying $T(I,u,v,F) = N(u,v)$ for each pair of words $(u,v) \in \Sigma^* \times \Delta^*$.*

4 Composition of Petri Net Transducers

In practical applications, it is important to be able to create complex transducers through composition of simple ones. To this end we lift the FST standard composition operations of union, product, closure and parallel product to clean PNTs (the parallel product is a new operation which cannot be applied to FSTs but is natural in case of PNTs). Cleanness ensures that runs always terminate properly and is shown to be preserved by the above operations. Moreover, we show that these composition operations preserve equivalence of PNTs.

Finally, we consider the central transducer composition operation of language composition. While the adaption of the previously mentioned composition operations for PNTs is more or less straightforward, there are several different possible adaptions of FST language composition.

In the following we consider a fixed concurrent semiring $\mathscr{S} = (S,\oplus,\otimes,\boxtimes,\overline{0},\overline{1})$.

4.1 Union, Sequential Product, Closure, Parallel Product

In this subsection we briefly lift the standard FST composition operations of union, (sequential) product and closure to clean PNTs and additionally define the parallel product of clean PNTs. All definitions and constructions are in the spirit of the FST case and rather straightforward. For each operation, first a functional definition is given defining the output weight of the composed PNT based on the output weights of the original PNTs and bisemiring-operations and generalising the corresponding FST definitions. Then an effective construction is given, showing that there is a composed PNT having the intended output weight. The constructions are illustrated in Figure 1, where for a compact presentation input symbols, output symbols and weights of transitions are omitted if possible. For the correspondence of the functional definitions and the constructions essentially theorem 2 can be used.

The *sum (or union)* $N_1 \oplus N_2$ of two PNTs N_1 and N_2 over $\mathscr{S}$ with the same input alphabet Σ and output alphabet Δ is defined as a PNT over $\mathscr{S}$ in such a way that for each pair of LPOs u over Σ and v over Δ:

$$(N_1 \oplus N_2)(u,v) = N_1(u,v) \oplus N_2(u,v).$$

The sum $N = N_1 \oplus N_2$ can be constructed in a straightforward way as the union of N_1 and N_2 together with additional new source and sink places and connecting transitions having empty input symbol, empty output symbol and weight $\overline{1}$. The construction yields that $WLPO(N_1 \oplus N_2)$ equals $WLPO(N_1) \cup WLPO(N_2)$ if events labelled by additional transitions are omitted.

The *product (concatenation)* $N_1 \otimes N_2$ of two PNTs N_1 and N_2 over $\mathscr{S}$ with the same input alphabet Σ and output alphabet Δ is defined as a PNT over $\mathscr{S}$ in such a way that for each pair of LPOs u over Σ and v over Δ:

$$(N_1 \otimes N_2)(u,v) = \bigoplus_{u=u_1;u_2,\, v=v_1;v_2} N_1(u_1,v_1) \otimes N_2(u_2,v_2).$$

The product of $n > 0$ instances of a PNT N we denote by N^n. By convention $N^0 = \mathscr{I}$, where $\mathscr{I}$ is the PNT satisfying $\mathscr{I}(u,v) = \overline{1}$ if u and v are both the empty LPO $(\emptyset,\emptyset,\emptyset)$ and $\mathscr{I}(u,v) = \overline{0}$ otherwise.

If N_1 is clean, the product $N = N_1 \otimes N_2$ can be constructed as the union of N_1 and N_2 together with a new transition having empty input symbol, empty output symbol and weight $\overline{1}$ and connecting the sink place of N_1 with the source place of N_2. The construction yields that $WLPO(N_1 \otimes N_2)$ equals $WLPO(N_1);WLPO(N_2)$ if events labelled by additional transitions are omitted.

The *closure* N^* of a PNT N over $\mathscr{S}$ with input alphabet Σ and output alphabet Δ is defined as PNT over $\mathscr{S}$ in such a way that for each pair of LPOs u over Σ and v over Δ:

$$N^*(u,v) = \bigoplus_{n=0}^{\infty} N^n(u,v).$$

If N is clean, $\bigoplus_{n=1}^{\infty} N^n(u,v)$ can be constructed by adding a transition with empty input label, empty output label and weight $\overline{1}$, which connects the sink place of N with

the source place of N and by adding additional new source and sink places and connecting transitions. Then N is the union of the resulting PNT and N^0. The construction yields that $WLPO(N^*)$ equals $WLPO(N)^*$ if events labelled by additional transitions are omitted.

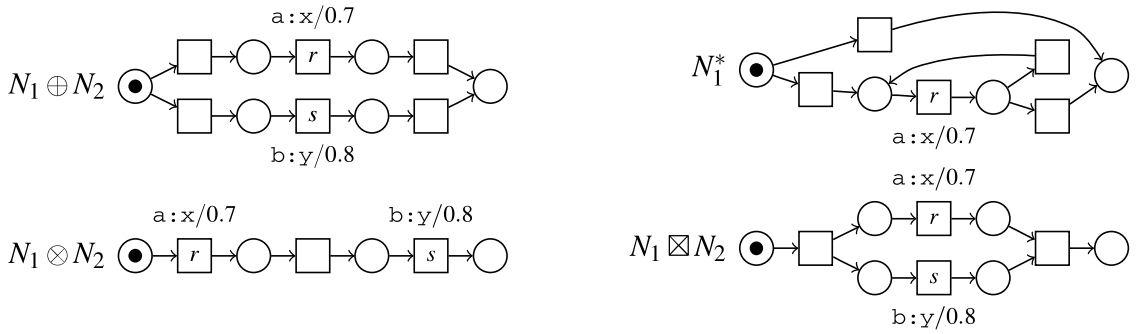

Fig. 1. Illustration of the union, (sequential) product, closure and parallel product of PNTs $N_1 = N(a,x,.7)$ and $N_2 = N(b,y,.8)$ over $\mathscr{V}$

The *parallel product* $N_1 \boxtimes N_2$ of two PNTs N_1 and N_2 over $\mathscr{S}$ with the same input alphabet Σ and output alphabet Δ is defined as a PNT over $\mathscr{S}$ in such a way that for each pair of LPOs u over Σ and v over Δ:

$$(N_1 \boxtimes N_2)(u,v) = \bigoplus_{u=u_1\|u_2, v=v_1\|v_2} N_1(u_1,v_1) \boxtimes N_2(u_2,v_2).$$

The parallel product $N = N_1 \boxtimes N_2$ can be constructed as the union of N_1 and N_2 together with additional new source and sink places and appropriate connecting transitions having empty input symbol, empty output symbol and weight $\overline{1}$. The construction yields that $WLPO(N_1 \boxtimes N_2)$ equals $WLPO(N_1) \parallel WLPO(N_2)$ if events labelled by additional transitions are omitted.

Theorem 3 (Composition Operations Preserve Cleanness). *Let N_1 and N_2 be clean PNTs. Then the PNTs $N_1 \oplus N_2$, $N_1 \otimes N_2$, N_1^* and $N_1 \boxtimes N_2$ are also clean.*

Proof. Clear by construction. □

An important application of equivalence in practise is the transformation of a PNT into an equivalent and simpler PNT. A central technique to do this is to replace parts of a complex composed PNT by equivalent parts. This technique requires that equivalence is consistent with composition operations.

Theorem 4 (Composition Operations Preserve Equivalence). *If N_1, N_1' and N_2, N_2' are (structure, output) equivalent, then also $N = N_1 \operatorname{op} N_2$ and $N' = N_1' \operatorname{op} N_2'$ with $\operatorname{op} \in \{\oplus, \otimes, \boxtimes\}$, as well as $N = N_1^*$ and $N' = (N_1')^*$ are (structure, output) equivalent.*

Proof. (sketch) As previously argued, in each case the set $WLPO(N,u,v)$ can be constructed from the sets $WLPO(N_1,u,v)$ and $WLPO(N_2,u,v)$ using some additional transitions which do not influence input label, output label and weight. Moreover, $N(u,v)$ can be computed from $N_1(u,v)$ and $N_2(u,v)$ through bisemiring operations.

We deduce that $WLPO(N,u,v) \neq \emptyset \Leftrightarrow WLPO(N',u,v) \neq \emptyset$, if $WLPO(N_i,u,v) \neq \emptyset \Leftrightarrow WLPO(N_i',u,v) \neq \emptyset$ for $i = 1,2$. If additionally $N_i(u,v) = N_i'(u,v)$ for $i = 1,2$, then also $N(u,v) = N'(u,v)$. □

4.2 Language Composition

In this subsection we present possibilities for lifting language composition of FSTs to PNTs. Basically, there are two possibilities for such a lifting. The first one is to generalise the construction from the FST case to the PNT case. As it will turn out, such a construction permits a functional definition only in the restricted case of doubled semirings. The second one is to adapt the functional definition from the FST case to the PNT case. Unfortunately, a concrete construction is possible only w.r.t. "weak" adaptions resp. in restricted cases.

Throughout this subsection we consider a PNT N_1, resp. FST T_1, over $\mathscr{S}$ with input alphabet Σ_1 and output alphabet Δ_1 and a PNT N_2, resp. FST T_2, over $\mathscr{S}$ with input alphabet $\Sigma_2 = \Delta_1$ and output alphabet Δ_2. In the FST case, the language composition $T_1 \circ T_2$ of T_1 and T_2 is essentially constructed from the Cartesian product of the two sets of states, the transitions of T_1 without output, the transitions of T_2 without input and transitions which are merged from a transition t_1 of T_1 and a transition t_2 of T_2 such that the output of t_1 equals the input of t_2, where the weights of t_1 and t_2 are multiplied. Finally, transitions which are not merged are put into an arbitrary but fixed sequence, i.e. weights are sequentially multiplied. This way, if $\otimes$ is commutative, $T_1 \circ T_2$ yields the functional equation

$$(T_1 \circ T_2)(u,w) = \bigoplus_v T_1(u,v) \otimes T_2(v,w),$$

where the sum runs over all words v over $\Sigma_2 = \Delta_1$ representing both output labels of paths of T_1 and input labels of paths of T_2.

We start with a generalisation of the FST construction to PNTs. The FST construction corresponds to the parallel product of N_1 and N_2 and merging each transition t_1 from N_1 with each transition t_2 from N_2 satisfying $\delta(t_1) = \sigma(t_2)$ to a transition t with input symbol $\sigma(t) = \sigma(t_1)$ and output symbol $\delta(t) = \delta(t_2)$, weight $\omega(t) = \omega(t_1) \otimes \omega(t_2)$ and connections ${}^\bullet t = {}^\bullet t_1 + {}^\bullet t_2$ and $t^\bullet = t_1^\bullet + t_2^\bullet$. Moreover, we keep all transitions of N_1 having empty output symbol, as well as all transitions of N_2 having empty input symbol (with unchanged input symbols, output symbols, weights and connections). All other transitions of N_1 or N_2 are omitted. We denote the constructed PNT by $N_1[\otimes]N_2$. If $\boxtimes$ is used to define the weight of merged transitions, we denote the resulting PNT by $N_1[\boxtimes]N_2$. Figure 2 illustrates the construction.

Theorem 5. *The PNT $N_1[\cdot]N_2$ satisfies the following properties:*

(i) If N_1 and N_2 are clean, then also $N_1[\cdot]N_2$ is clean.

(ii) If $\mathscr{S}$ is the doubled semiring, then $(N_1[\otimes]N_2)(u,w) = \bigoplus_v N_1(u,v) \otimes N_2(v,w)$, where the sum runs over all LPOs v over $\Sigma_2 = \Delta_1$ representing output labels of weighted LPO-runs of N_1 and input labels of weighted LPO-runs of N_2. In particular $N_1[\otimes]N_2$ generalises FST language composition.

(iii) *If* $(N_1[\otimes]N_2)(u,w) = \bigoplus_v N_1(u,v)\,\mathrm{op}\,N_2(v,w)$*, where the sum runs over all LPOs* v *over* $\Sigma_2 = \Delta_1$ *representing output labels of weighted LPO-runs of* N_1 *and input labels of weighted LPO-runs of* N_2 *and* op *is a semiring operation, then* $\mathrm{op} = \otimes$ *and* $\mathscr{S}$ *is the doubled semiring.*

(iv) *If* N_1, N_1' *and* N_2, N_2' *are structure equivalent, then also* $N = N_1[\cdot]N_2$ *and* $N' = N_1'[\cdot]N_2'$ *are structure equivalent.*

(v) *If* N_1, N_1' *and* N_2, N_2' *are output equivalent, then* $N = N_1[\cdot]N_2$ *and* $N' = N_1'[\cdot]N_2'$ *need not be output equivalent.*

Proof. (sketch)

ad (i): By construction, every reachable marking m of $N_1[\cdot]N_2$ – except for the initial and final marking – is of the form $m = m_1 + m_2$ for some reachable marking m_1 of N_1 and some reachable marking m_2 of N_2. Since both N_1 and N_2 are clean the only reachable marking which marks the sink places $p_{1,F}$ and $p_{2,F}$ of N_1 and N_2 is $p_{1,F} + p_{2,F}$. The only transition step which can occur in this marking leads to the new sink place of $N_1[\cdot]N_2$. It follows that $N_1[\cdot]N_2$ is clean.

ad (ii): Follows from $\omega_{sp}(wlpo) = \bigotimes_{x \in V} \nu(x)$ for WLPOs $wlpo = (V, <, l, \nu)$.

ad (iii): Consider $N_1 = N(x,y,a) \boxtimes N(u,v,b)$ and $N_2 = N(y,z,c) \boxtimes N(v,w,d)$. Then $(a \otimes c) \boxtimes (b \otimes d) = (N_1[\otimes]N_2)(x \parallel u, z \parallel w) = N_1(x \parallel u, y \parallel v)\,\mathrm{op}\,N_2(y \parallel v, z \parallel w) = (a \boxtimes b)\,\mathrm{op}\,(c \boxtimes d)$. Putting $b = d = \overline{1}$ yields $a\,\mathrm{op}\,c = a \otimes c$. Putting $a = d = \overline{1}$ yields $b \otimes c = b \boxtimes c$.

ad (iv): Follows from $L(N_1[\cdot]N_2) = \{(u,w) \mid \exists v : (u,v) \in L(N_1), (v,w) \in L(N_2)\}$. This can be seen as follows: If $wlpo \in L(N_1[\cdot]N_2, u, w)$, then the restriction to nodes which are labelled by transitions from N_i and by merged transitions and the relabelling of merged transition labels by corresponding transition labels from N_i yields WLPO-runs $wlpo_1 \in L(N_1, u, v_1)$ and $wlpo_2 \in L(N_2, v_2, w)$ with $v_1 = v_2$. If on the other side $wlpo_1 \in L(N_1, u, v)$ and $wlpo_2 \in L(N_2, v, w)$, then $wlpo \in L(N_1[\cdot]N_2, u, w)$ can be constructed from the union of $wlpo_1$ and $wlpo_2$ through merging events corresponding to merged transitions of $N_1[\cdot]N_2$.

ad (v): Consider the PNTs $N_1 = N(a,x,.4) \boxtimes N(b,y,.8)$, $N_1' = N(a,x,.8) \boxtimes N(b,y,.4)$ and $N_2 = N(x,r,.5) \boxtimes N(y,s,1)$ over the concurrent semiring $\mathscr{V}$. Then N_1 and N_1' are output equivalent, but $N_1[\otimes]N_2$ and $N_1'[\otimes]N_2$ are not. In a similar way, using sequential composition of PNTs, a counterexample for $N_1[\boxtimes]N_2$ can be found. □

Remark 1. In case of part (ii) of the previous theorem the construction for the language composition of two PNTs is different and much simpler than the construction in the case of FSTs. In particular, if N_1 and N_2 are PNTs representing FSTs T_1 and T_2, then $N_1[\otimes]N_2$ does not represent an FST, since it is no state machine. Namely, transitions of N_1 with empty output and transitions of N_2 with empty input may be concurrent. Nevertheless $N_1[\otimes]N_2$ is output equivalent to the PNT representing $T_1 \circ T_2$ on pairs of sequences (u,v).

We now consider possible adaptions of the functional definition of FST language composition. A direct adaption yields

$$(N_1 \circ N_2)(u,w) = \bigoplus_{v \in L_O(N_1) \cap L_I(N_2)} N_1(u,v) \otimes N_2(v,w). \tag{1}$$

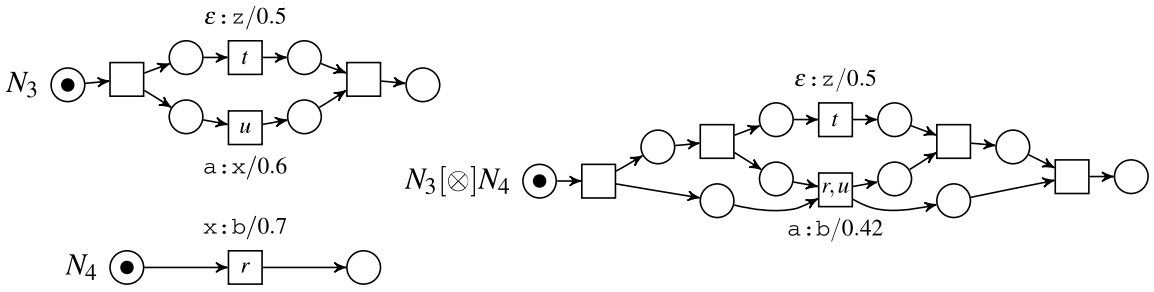

Fig. 2. Language composition for $N_3 = N(x,b,.6) \boxtimes N(\varepsilon,z,.5)$ and $N_4 = N(a,x,.7)$ over $\mathscr{Y}$

Our belief is, that there is no construction satisfying this functional equation within the considered framework of PT-net based PNTs over general concurrent semirings. A proof of such a result is topic of further research, but there are constructions for some relaxations of equation 1.

Theorem 6. *It holds:*

(i) There is a PNT N with $N(u,w) = \bigoplus_{v,v' \in L_O(N_1) \cap L_I(N_2) : l(v) = l(v')} N_1(u,v) \otimes N_2(v',w)$, where $l(v)$ is the multiset of transition labels of an LPO v. The construction of N preserves structure and output equivalence, but not cleanness.

(ii) If the state graph of $N_1[\cdot]N_2$ is acyclic, then there is a PNT N satisfying $N(u,w) = \bigoplus_{v,v' \in L_O(N_1) \cap L_I(N_2) : v \leq v' \vee v' \leq v} N_1(u,v) \otimes N_2(v',w)$, where $v \leq v'$ means that v is an extension of v'. The construction of N preserves structure and output equivalence and cleanness. In particular N generalises FST language composition.

Proof. (sketch)

The basic idea for the constructions is to consider the sequential product of two PNTs N_1' and N_2', such that $L_O(N_1') = L_O(N_1) \cap L_I(N_2)$ and N_1' behaves like N_1 on its output words, and N_2' has similar properties. The sequential product of N_1' and N_2' satisfies

$$(N_1' \otimes N_2')(u,w) = \bigoplus_{v,v' \in L_O(N_1) \cap L_I(N_2)} N_1(u,v) \otimes N_2(v',w).$$

It is easy to observe that such N_1' and N_2' both can be constructed from $N_1[\cdot]N_2$, such that N_1' and N_2' have isomorphic underlying PT-nets, but different input and output symbols and weights, such that N_1' reads u but writes nothing and N_2' reads nothing but writes w w.r.t. the same LPO-run. Since $\overline{1}$ is the unit of both multiplications, this can be done for N_1' (and similar for N_2') as follows:

- Each transition gets ε as output symbol.
- Each transition t merged from t_1 of N_1 and t_2 of N_2 gets the weight of t_1.
- All transitions coming from N_2 get weight $\overline{1}$.

ad (i): Through adding additional places connecting transitions of N_1' with their isomorphic images in N_2' it is possible to restrict the behaviour in such a way that after executing an LPO-run of N_1' only LPO-runs of N_2' having the same set of transition occurrences can be executed. If the state graph of $N_1[\cdot]N_2$ contains cycles, the construction

does not preserve cleanness, since additional places may be unbounded. The preservation of equivalences follows since $L(N)$ can be constructed from $L(N_1)$ and $L(N_2)$ and $N(u,w)$ is computed from $N_1(u,v)$ and $N_2(v',w)$ through bisemiring operations.

ad (ii): In the special case of an acyclic state graph, it is possible to add transition copies and places ensuring that transitions, which are in structural conflict in N_1' and which are executed in a specific order within an LPO-run of N_1', are executed in the same order afterwards in N_2'. Moreover, cleanness is preserved. The preservation of equivalences follows similar to (i).

□

Remark 2. Note that $l(v) = l(v')$ but $v \neq v'$ is possible. For example, the PNT $(N(a,b,\omega) \oplus N(c,d,\nu))^*$ allows the outputs bd and db. This means, there are different LPO-runs of a PNT with equal sets of transition occurrences if there are transitions which are in structural conflict, but occur together within an LPO-run due to cycles in the PNT. The equation mentioned in property (i) of theorem 6 is not a generalisation of the FST case.

5 Conclusion and Outlook

In this paper we introduced weighted Petri net transducers for the translation of partial languages. As a central result we have shown that the used weight structure of concurrent semirings is the least restrictive idempotent bisemiring structure such that partial words with fewer dependencies have bigger weights. Moreover, we defined composition operations of union, product, closure, parallel product and language composition on PNTs which preserve equivalence of PNTs. The output weights of a composed PNT can be computed from the output weights of its components using bisemiring operations, where in the case of language composition such a compositional computation is possible only in restricted cases.

There are important further steps in several directions. Due to lack of space we only mention here that there are still several central aspects of the framework of FSTs which need to be examined also w.r.t. PNTs, for example additional composition operations (inversion, reversal), optimisation algorithms (ε-elimination, weight-pushing) and on-the-fly simulation algorithms.

References

1. Azéma, P., Balbo, G. (eds.): ICATPN 1997. LNCS, vol. 1248. Springer, Heidelberg (1997)
2. Best, E., Devillers, R.R., Hall, J.G.: The box calculus: a new causal algebra with multi-label communication. In: Rozenberg [19], pp. 21–69
3. Boudol, G., Castellani, I.: On the semantics of concurrency: Partial orders and transition systems. In: Ehrig, H., Levi, G., Montanari, U. (eds.) CAAP 1987 and TAPSOFT 1987. LNCS, vol. 249, pp. 123–137. Springer, Heidelberg (1987)
4. Chothia, T., Klejin, J.: Q-automata: Modelling the resource usage of concurrent components. Electronic Notes in Theoretical Computer Science 175(175), 153–167 (2007)
5. Droste, M., Kuich, W.: Semirings and Formal Power Series. In: Droste, et al. (eds.) [6], ch.1, pp. 3–28 (2009)

6. Droste, M., Kuich, W., Vogler, H. (eds.): Handbook of Weighted Automata. Monographs in Theoretical Computer Science. Springer (2009)
7. Esposito, A., Esposito, A.M., Vinciarelli, A., Hoffmann, R., Müller, V.C. (eds.): COST 2102. LNCS, vol. 7403. Springer, Heidelberg (2012)
8. Fichtner, I., Kuske, D., Meinecke, I.: Traces, Series-Parallel Posets, and Pictures: A Weighted Study. In: Droste, et al. (eds.) [6], ch. 10, pp. 405–452 (2009)
9. Füllöp, Z., Vogler, H.: Weighted Tree Automata and Tree Transducers. In: Droste, et al. (eds.) [6], ch. 9, pp. 313–404 (2009)
10. Gischer, J.L.: The equational theory of pomsets. Theoretical Computer Science 61, 199–224 (1988)
11. Grabowski, J.: On Partial Languages. Fundamenta Informaticae 4(2), 428–498 (1981)
12. Hack, M.: Petri net languages. Technical Report Memo 124, computation structures group, massachusetts institute of technology (1975)
13. Hoare, T., Möller, B., Struth, G., Wehrman, I.: Concurrent Kleene algebra and its foundations. The Journal of Logic and Algebraic Programming 80, 266–296 (2011)
14. Kuske, D., Meinecke, I.: Branching automata with costs - a way of reflecting parallelism in costs. Theoretical Computer Science 328, 53–75 (2004)
15. Lorenz, R., Huber, M.: Petri net transducers in semantic dialogue modelling. In: Proceedings of “Elektronische Sprachsignalverarbeitung (ESSV)". Studientexte zur Sprachkommunikation, vol. 64, pp. 286–297 (2012)
16. Lorenz, R., Huber, M.: Realizing the Translation of Utterances into Meanings by Petri Net Transducers. In: Proceedings of “Elektronische Sprachsignalverarbeitung (ESSV)". Studientexte zur Sprachkommunikation, vol. 65 (2013)
17. Mohri, M.: Weighted Automata Algorithms. In: Droste, et al. (eds.) [6], ch. 6, pp. 213–254 (2009)
18. Pratt, V.: Modelling Concurrency with Partial Orders. Int. Journal of Parallel Programming 15, 33–71 (1986)
19. Rozenberg, G. (ed.): Advances in Petri Nets 1992, The DEMON Project. Springer (1992)
20. Straßner, D.: Prototypische Implementierung von Petrinetz-Transduktoren mit SNAKES. Bachelor thesis, Augsburg University (2013)
21. van Biljon, W.R.: Extending Petri nets for specifying man-machine dialogues. Int. J. Man-Mach. Stud. 28(4), 437–455 (1988)
22. van der Aalst, W.M.P.: Verification of workflow nets. In: Azéa, Balbo (eds.) [1], pp. 407–426
23. Wang, F.-Y., Mittmann, M., Saridis, G.N.: Coordination specification for CIRSSE robotic platform system using Petri net transducers. Journal of Intelligent and Robotic Systems 9, 209–233 (1994)
24. Wang, F.-Y., Saridis, G.N.: A model for coordination of intelligent machines using Petri nets. In: Proceedings of the IEEE International Symposium on Intelligent Control, pp. 28–33. IEEE Comput. Soc. Press (1989)
25. Wirsching, G., Huber, M., Kölbl, C.: Zur Logik von Bestenlisten in der Dialogmodellierung. In: Proceedings of “Elektronische Sprachsignalverarbeitung (ESSV)". Studientexte zur Sprachkommunikation, vol. 61, pp. 309–316 (2011)
26. Wirsching, G., Huber, M., Kölbl, C., Lorenz, R., Römer, R.: Semantic dialogue modeling. In: Esposito, et al. (eds.) [7], pp. 104–113
27. Wolff, M.: Akustische Mustererkennung. Habilitation (2009)