



On the Generation of Dossier Data Structures for Workflow Nets

Kees van Hee⁽¹⁾, Jan Hidders⁽²⁾, Geert-Jan Houben⁽³⁾, Jan Paredaens⁽²⁾, Philippe Thiran⁽⁴⁾

(1) Eindhoven University of Technology (TU/e)

(2) University of Antwerp (UA)

(3) Vrije Universiteit Brussel (VUB)

(4) University of Namur (FUNDP)



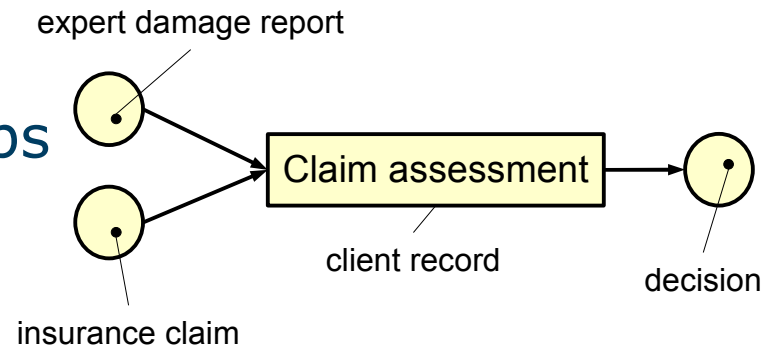
Outline

- Motivation
- Workflow nets
- Jackson types
- Jackson nets
- Relationships between J. types and J. nets
- Characterization of Jackson nets



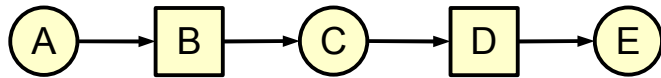
Motivation

- Storing all information that was involved in a complete run of the workflow
- For each step:
 - inputs from preceding steps
 - any external inputs
 - produced outputs
- What would be a convenient and intuitive data structure for storing (and querying etc.) this?
 - “Dossier data structures”

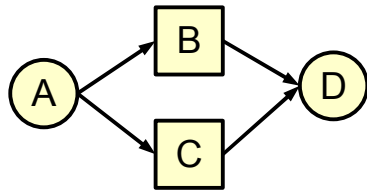




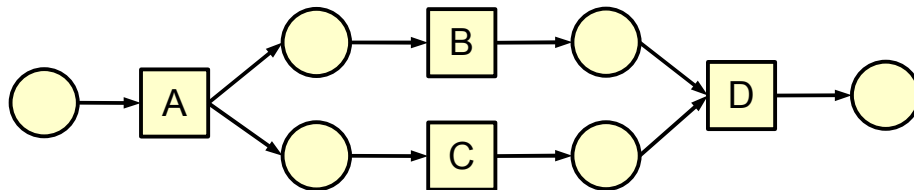
A few Easy Cases



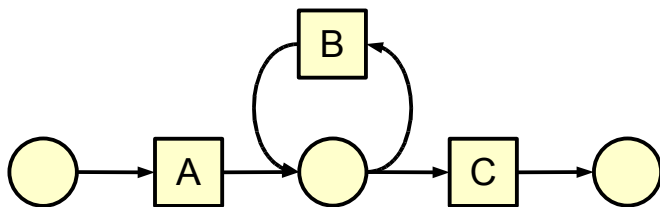
(A, B, C, D, E) where
 $(..)$ denotes **ordered** tuple type



$(A, \langle B, C \rangle, D)$ where
 $\langle .. \rangle$ denotes labeled union type



$(A, \{B, C\}, D)$ where $\{..\}$
denotes **unordered** tuple type



$(A, [B], C)$ where $[..]$
denotes a list type



Research questions

- How can we generate for Workflow nets such types consisting of
 - ordered tuples,
 - unordered tuples,
 - labeled unions and
 - lists?
- For which class of Workflow nets is this possible?



Workflow Nets (1/3)

- Workflow nets are subclass of Petri nets:
 - special **input place** p^i with no incoming edge
 - special **output place** p^o with no outgoing edges
 - strongly connected if edge from p^o to p^i is added
- In addition places and transitions are labeled
 - label denotes the data structure of event
- Runs are defined as usual
 - **initial marking**: 1 token in p^i
 - **final marking**: 1 token in p^o
 - **full run**: run from p^i to p^o



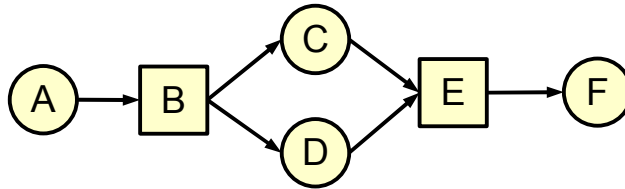
Workflow Nets (2/3)

- Soundness:
 - from all markings reachable from initial marking we can reach the final marking
 - all transitions are fired in at least one full run
- Trace set:
 - labels are assumed to be atomic types / events
 - **associated net** is net where places are replaced with transitions with the same label
 - trace consists of the sequence of labels of firing transitions in a full run of the associated net

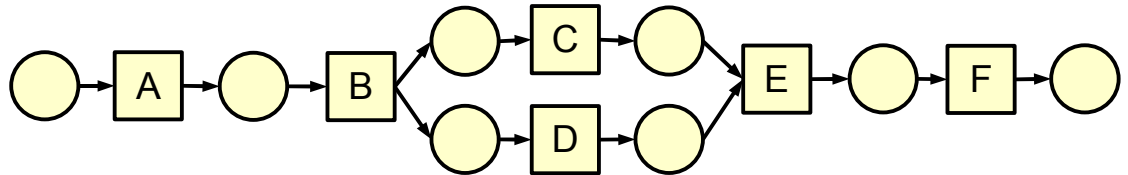


Workflow Nets (3/3)

workflow net: Ω



associated net: Ω'



trace set: $\text{Tr}(\Omega) = \{ \text{ABCDEF}, \text{ABDCEF} \}$

Alternative formulation:

- tokens can be either inactive or active
 - inactive right after production, active after activation
 - event is either firing of a transition (event is label of transition) or activation of a token in a place (event is label of place)
- transitions can only fire if all required tokens are active

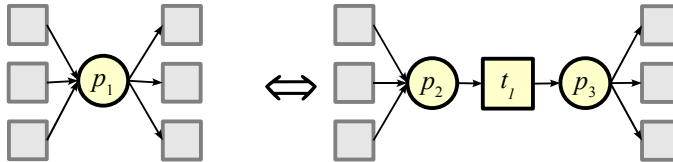


- General types:
 - $J ::= \mathbf{A} \mid (J;J) \mid (J\parallel J) \mid (J+J) \mid (J\#J)$
 - \mathbf{A} is postulated set of atomic types
 - $;$ is ordered tuple constructor / concatenation
 - \parallel is unordered tuple constructor / parallel
 - $+$ is labeled union constructor / choice
 - $\#$ is separated list constructor / iteration
 - $A\#B \equiv A;(B;A)^*$
- Trace set of type τ : $\text{Tr}(\tau)$
 - strings of atomic types, in obvious way
- Trace equivalence:
 - $\tau_1 \equiv_{\text{tr}} \tau_2$ iff $\text{Tr}(\tau_1) = \text{Tr}(\tau_2)$



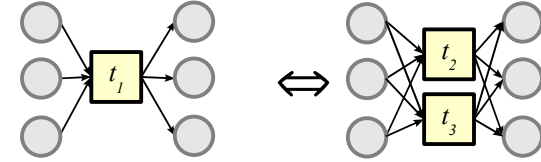
Jackson Nets (1/3)

R1: Sequential place split:



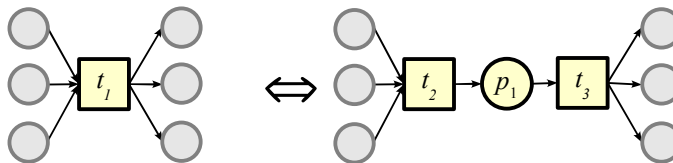
$$\lambda(p_1) = (\lambda(p_2) ; (\lambda(t_1) ; \lambda(p_3)))$$

R5: OR split:



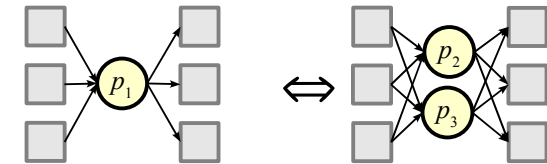
$$\lambda(t_1) = (\lambda(t_2) + \lambda(t_3))$$

R2: Sequential transition split:



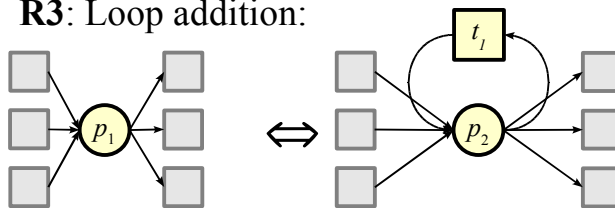
$$\lambda(t_1) = (\lambda(t_2) ; (\lambda(p_1) ; \lambda(t_3)))$$

R4: AND split:



$$\lambda(p_1) = (\lambda(p_2) \parallel \lambda(p_3))$$

R3: Loop addition:



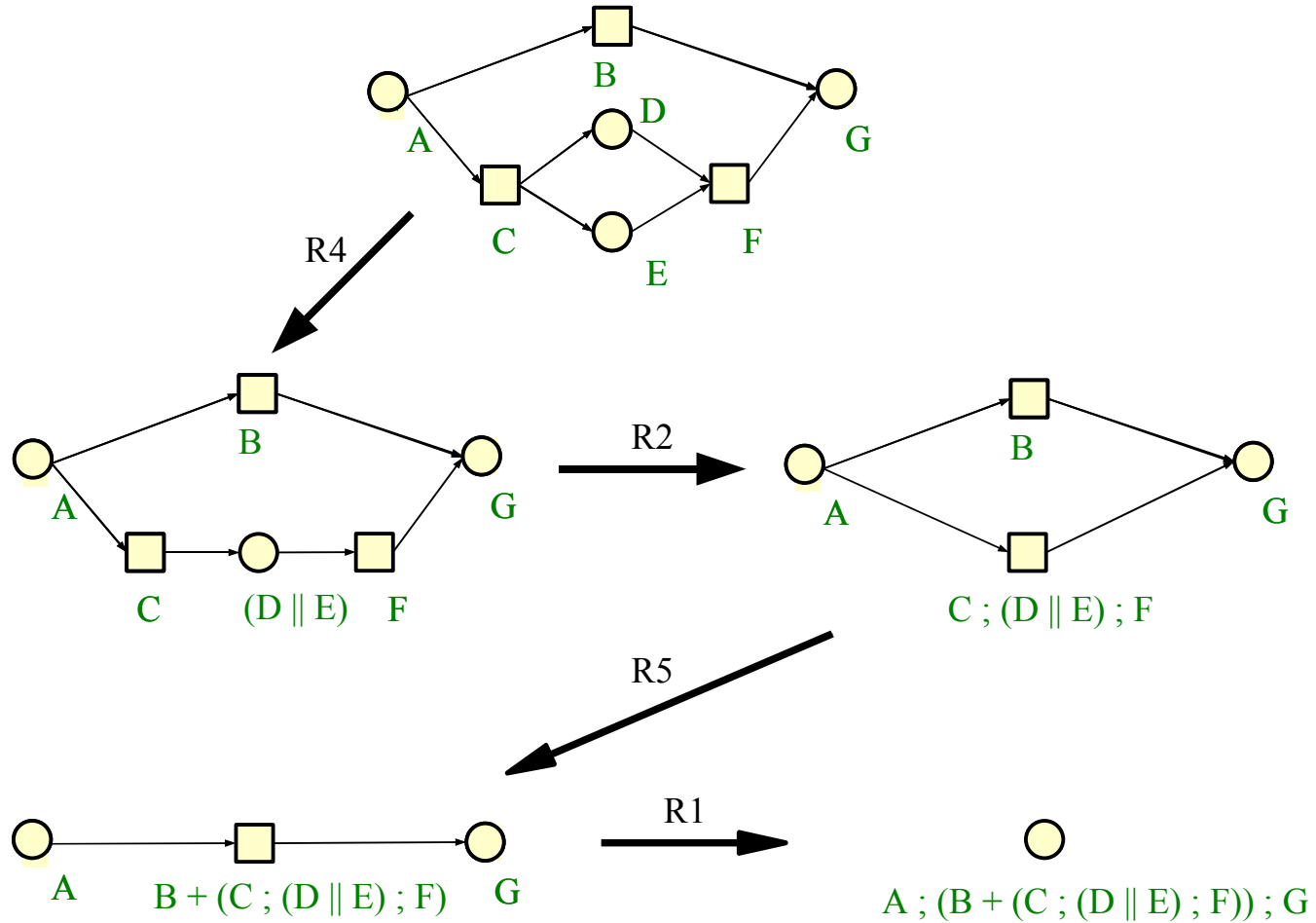
$$\lambda(p_1) = (\lambda(p_2) \# \lambda(t_1))$$

- Reducing WF net to a single place generates a type

Rules independently established by
Piotr Chrzastowski-Wachtel et al, BPM 2003



Jackson Nets (2/3)





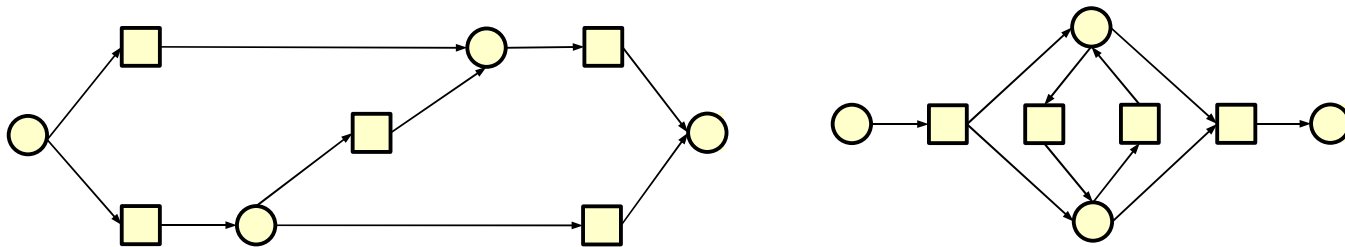
Jackson Nets (3/3)

- Process can be reversed:
 - start from a single place labeled with a type
 - apply rules until all labels are atomic types
- Works for subset of types called **Jackson types**:
 - $J^o ::= \mathbf{A} \mid (\mathbf{A};(J^t;\mathbf{A}))$
 - $J^t ::= \mathbf{A} \mid (J^t;(J^p;J^t)) \mid (J^t+J^t)$
 - $J^p ::= \mathbf{A} \mid (J^p;(J^t;J^p)) \mid (J^p\parallel J^p) \mid (J^p\#J^t)$
- Generates always a workflow net: **Jackson nets**
 - other types may also generate workflow nets
 - but Jackson nets are those WF nets with atomic labels for which we can generate a type with the given rules
- Then J. net is said to be **associated with** J. type
 - many-to-many relationship



Properties of Jackson Nets

- **Theorem 1** All Jackson nets are sound workflow nets, but not all sound workflow nets are Jackson nets.
 - proof sketch: with induction, if we apply a rule to a sound net we obtain again a sound net.
- Not Jackson nets:



- **Theorem 2** If a Jackson net Ω is associated with Jackson type τ then $\text{Tr}(\Omega) = \text{Tr}(\tau)$
 - establishes correctness of generated type
 - proof sketch: generalizing notion of trace set for nets labeled with non-atomic types



Jackson nets and J. types (1/2)

- To what extent is the generated type determined?
 - Algebraic equivalence: $\tau_1 \equiv_{\text{alg}} \tau_2$
 - $(\tau_1 ; \tau_2) ; \tau_3 \equiv_{\text{alg}} \tau_1 ; (\tau_2 ; \tau_3)$
 - $(\tau_1 \parallel \tau_2) \parallel \tau_3 \equiv_{\text{alg}} \tau_1 \parallel (\tau_2 \parallel \tau_3)$ $\tau_1 \parallel \tau_2 \equiv_{\text{alg}} \tau_2 \parallel \tau_1$
 - $(\tau_1 + \tau_2) + \tau_3 \equiv_{\text{alg}} \tau_1 + (\tau_2 + \tau_3)$ $\tau_1 + \tau_2 \equiv_{\text{alg}} \tau_2 + \tau_1$
 - $(\tau_1 \# \tau_2) \# \tau_3 \equiv_{\text{alg}} \tau_1 \# (\tau_2 + \tau_3)$
 - **Theorem 3** If a Jackson net Ω is associated with Jackson types τ_1 and τ_2 then $\tau_1 \equiv_{\text{alg}} \tau_2$
 - proof sketch: use algebraic rules to “normalize” type and show that associated normalized type is unique



Jackson nets and J. types (2/2)

- To what extent is the Jackson net determined by the Jackson type?
 - **Theorem 4** If Ω_1 and Ω_2 are associated with τ_1 and τ_2 , respectively, and $\tau_1 \equiv_{\text{alg}} \tau_2$ then Ω_1 and Ω_2 are isomorphic
 - proof sketch: Show with induction that abstract syntax tree of the type determines which nodes and edges there are. Show that algebraic rewrite does not change this.
 - **Corollary** If Ω_1 and Ω_2 associated with τ_1 and τ_2 , respectively, then the following are equivalent:
 - Ω_1 and Ω_2 are isomorphic
 - $\tau_1 \equiv_{\text{alg}} \tau_2$



Characterization of J. nets

- **Theorem 5** A sound WF net Ω with atomic type labels and no duplicate labels is a Jackson net iff there is a Jackson type τ such that each atomic type appears at most once and $\text{Tr}(\Omega) = \text{Tr}(\tau)$.
 - proof sketch: Jackson types can only describe the trace set of safe nets. For such nets the trace set tells us which events enable which other events, and therefore which edges there are in the net.
- **Extended corollary** If Ω_1 and Ω_2 have no duplicate labels and are associated with τ_1 and τ_2 , respectively, then the following are equivalent:
 - Ω_1 and Ω_2 are isomorphic
 - $\tau_1 \equiv_{\text{alg}} \tau_2$
 - $\text{Tr}(\Omega_1) = \text{Tr}(\Omega_2)$
 - $\text{Tr}(\tau_1) = \text{Tr}(\tau_2)$



Conclusion

- Jackson nets are interesting subset of sound WF nets for generation of dossier data types
 - One-to-one relationship between J. nets and J. types, up to some algebraic equivalence
- Algebraic identities axiomatize trace equivalence on Jackson types where every atomic type appears at most once
- Characterization of Jackson nets as nets with trace set of a Jackson type where every atomic type appears at most once.