

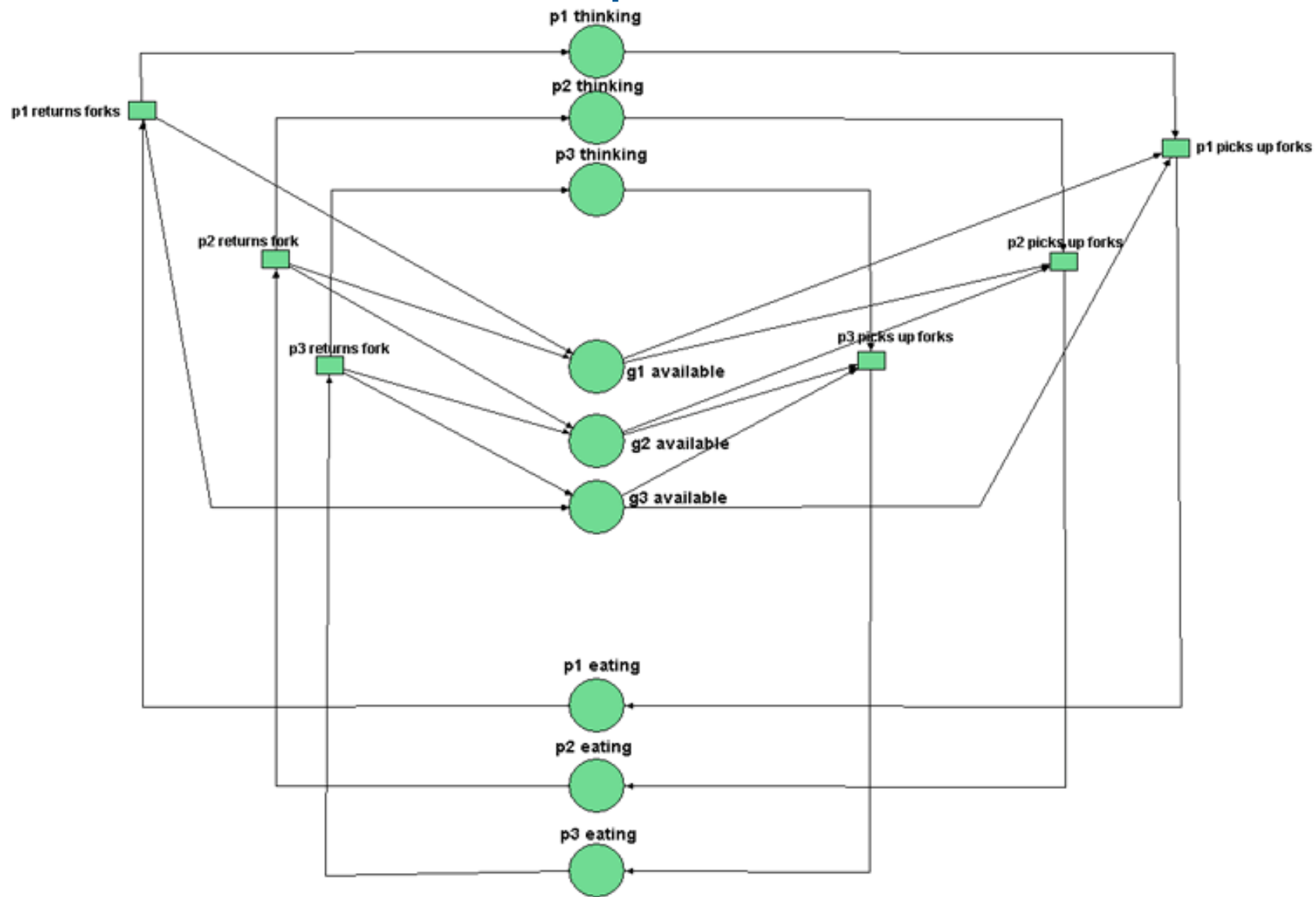
High-Level Petrinetze

SE Analyse von Petrinetzen

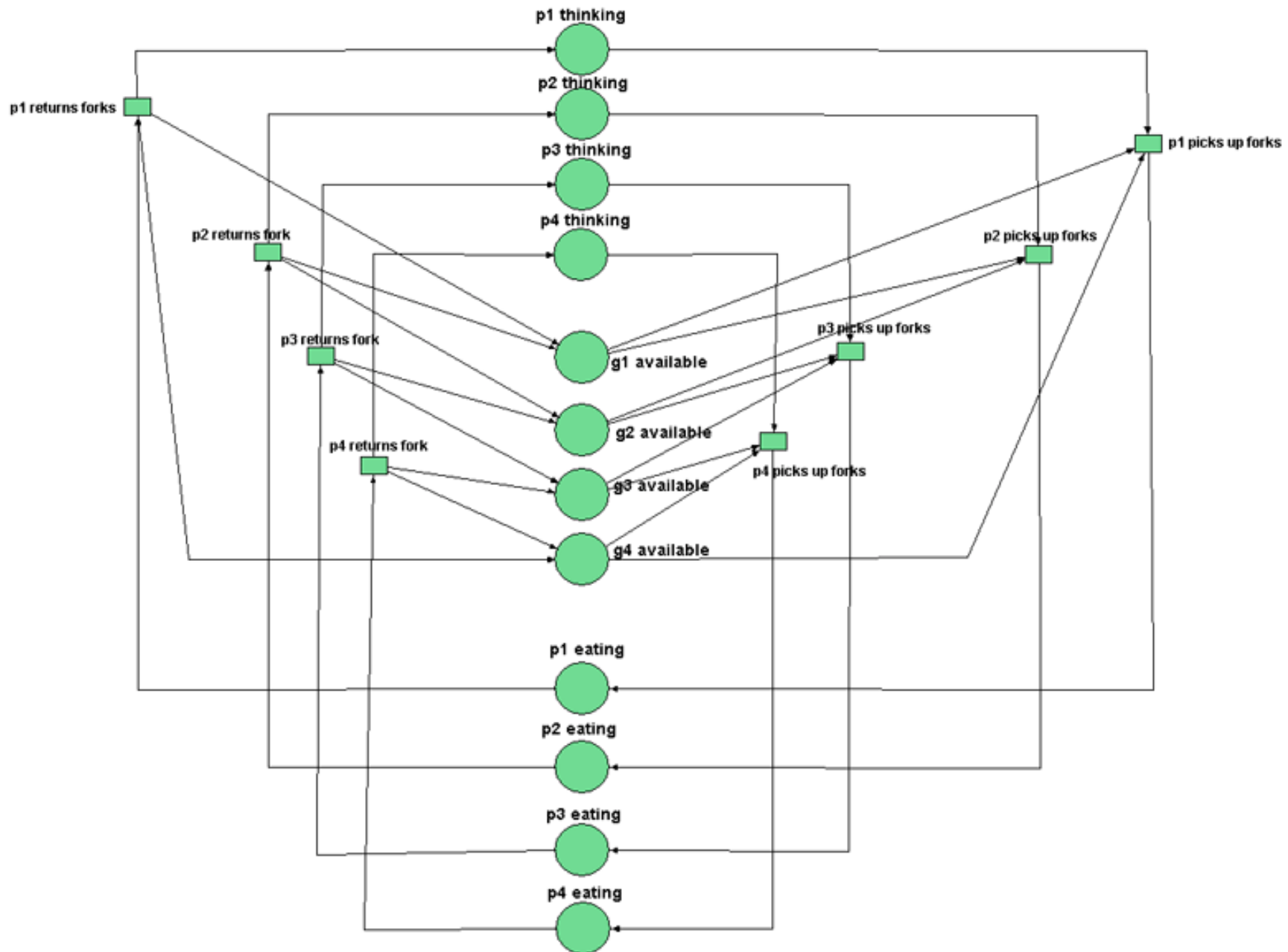
Jan Sürmeli

13.02.2008

Motivation: 3 Philosophen, Low Level



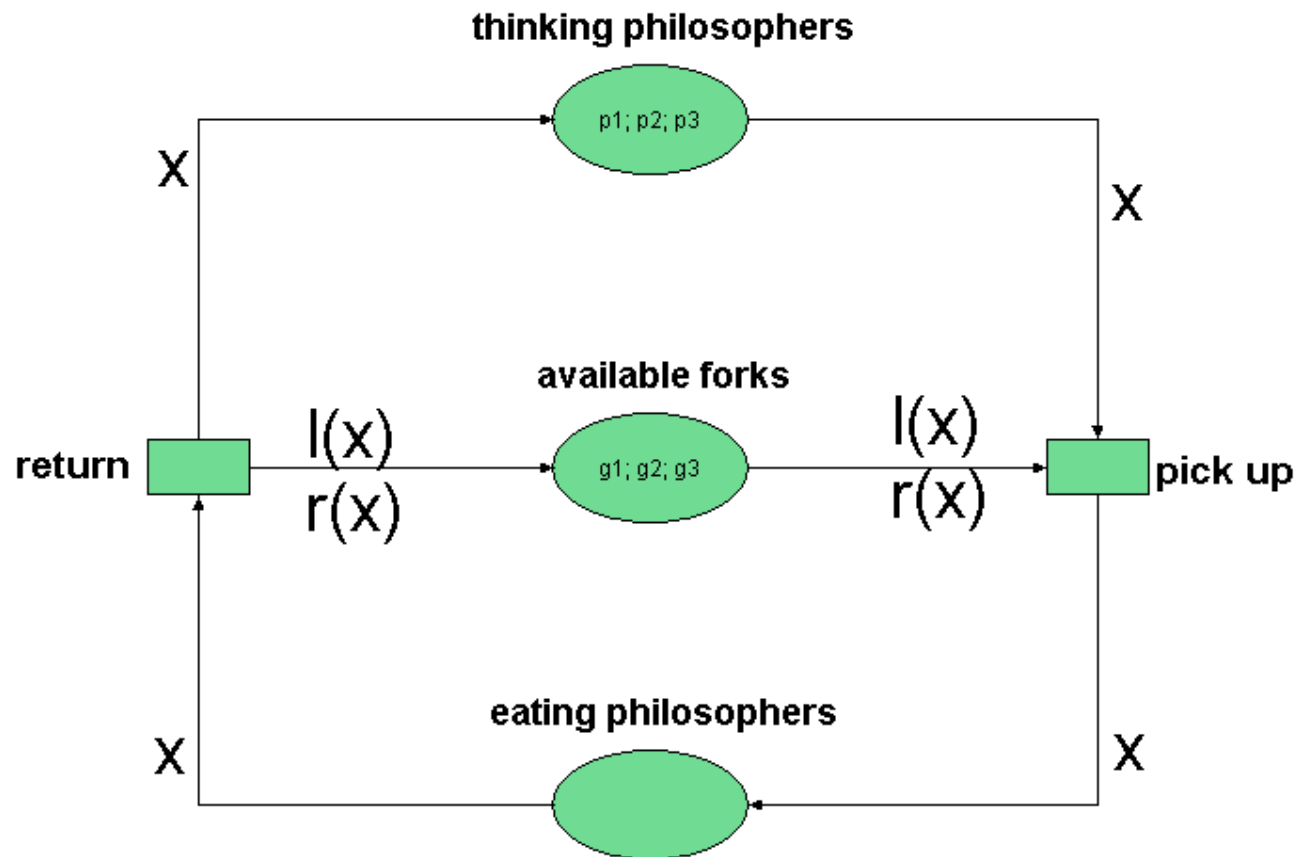
Motivation: 4 Philosophen, Low Level



Motivation: Ein Lösungsansatz

- Petrietze werden schnell sehr unübersichtlich
- Idee: Fasse „Dinge“ zusammen, die ähnlich sind
 - Im Beispiel:
 - Plätze: denkende Philosophen, essende Philosophen, verfügbare Gabeln
 - Transitionen: Gabeln aufnehmen, Gabeln zurückgeben
 - Voraussetzung: Unterscheidbare Marken

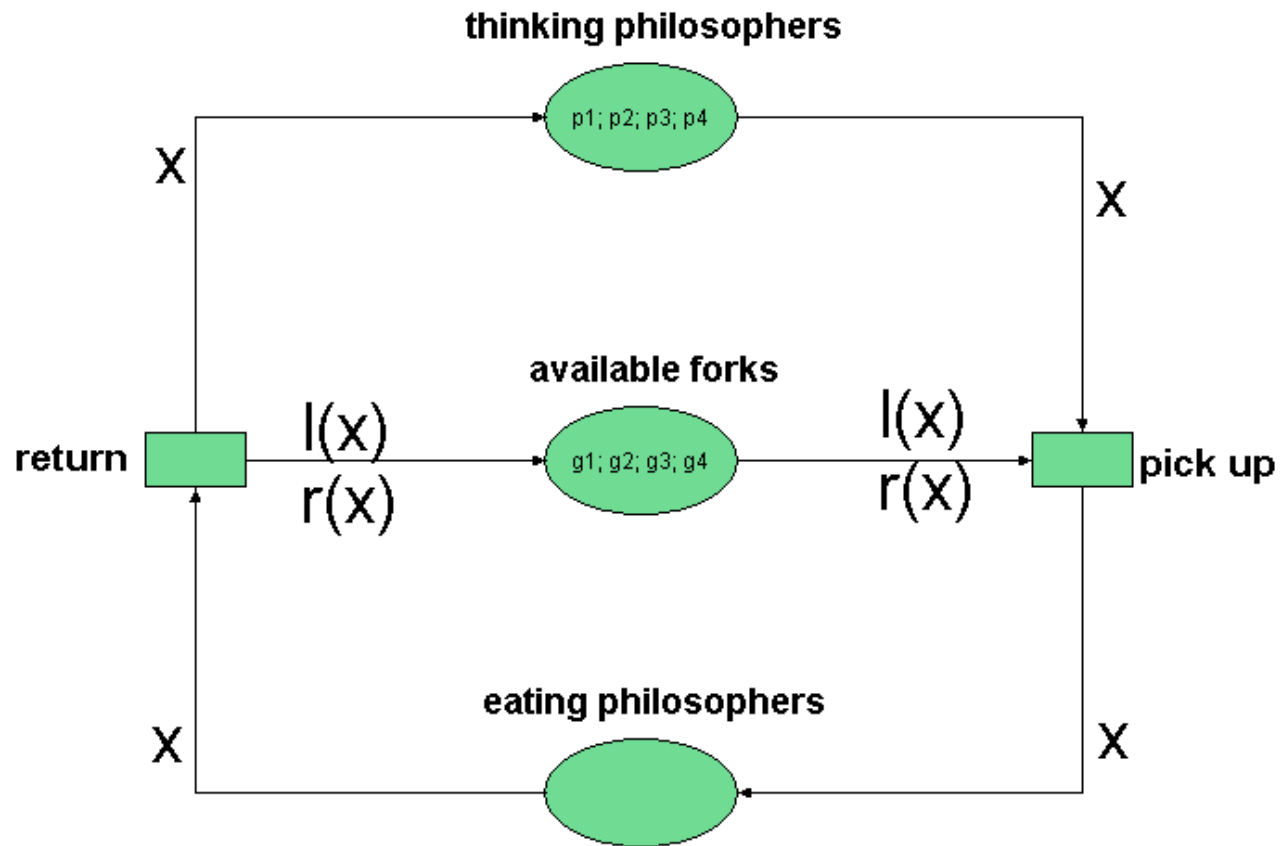
Neue Lösung: 3 Philosophen, High Level



$P = \{p_1, p_2, p_3\}$
 $G = \{g_1, g_2, g_3\}$
 $l, r : P \rightarrow G$
 x : variable over P

$l(p_1) = r(p_2) = g_1$
 $l(p_2) = r(p_3) = g_2$
 $l(p_3) = r(p_1) = g_3$

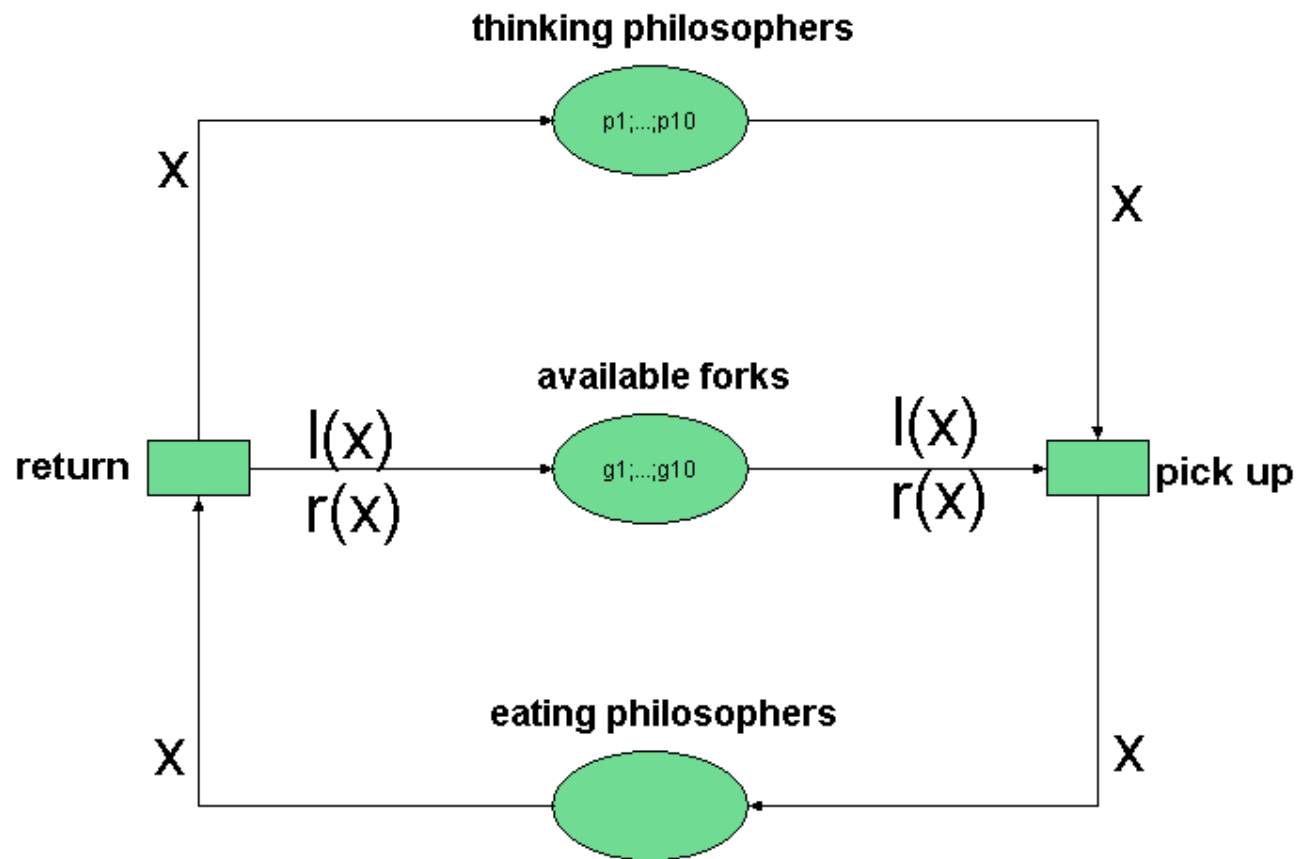
Neue Lösung: 4 Philosophen, High Level



$P = \{p1, p2, p3, p4\}$
 $G = \{g1, g2, g3, g4\}$
 $l, r : P \rightarrow G$
 x : variable over P

$l(p1) = r(p2) = g1$
 $l(p2) = r(p3) = g2$
 $l(p3) = r(p4) = g3$
 $l(p4) = r(p1) = g4$

Neue Lösung: 10 Philosophen, High Level



$P = \{p_1, \dots, p_{10}\}$
 $G = \{g_1, \dots, g_{10}\}$
 $l, r : P \rightarrow G$
 x : variable over P

$l(p_1) = r(p_2) = g_1$
...
 $l(p_{10}) = r(p_1) = g_{10}$

Veränderungen gegenüber Low Level

- Was ist anders?
 - „Farbige“ Token
 - Vorher: Nur schwarze Token
 - Jetzt: Unterscheidbare Token (Daten)
 - Beschriftungen
 - Vorher: Kantengewichtungen, Namen
 - Jetzt: Neue Kantenbeschriftungen

Effekt Low Level → High Level

- Was ist passiert?
 - Generalisierung
 - Vorher: Nur Kontrollstrukturen
 - Jetzt: Auch Datenstrukturen
 - Shorthand-Notation
 - Vorher: Viele ähnlich strukturierte Netze
 - Jetzt: Ähnliche Strukturen zusammengefasst

Analogie zu Programmiersprachen

- Low-Level-Programmiersprachen:
 - Maschinen oder Asemblysprachen, die keinen Compiler oder Intepreter benötigen
- High-Level-Programmiersprachen:
 - Jede Sprache, die erst kompiliert oder interpretiert werden muss
 - Quelltext ist im Allgemeinen:
 - Kürzer
 - Schöner
 - Lesbarer
 - Wiederverwendbarer

Die Notation und ihre Bedeutung

- Heute: Überblick, intuitiv und informal
- Wir betrachten das Schalten einer Transition
 - Deklaration von Sorten, Variablen, Funktionen
 - Plätze
 - Konsumieren
 - Transition
 - Produzieren
 - Effekt des Schaltvorgangs

Sorten und Variablen

- Was ist eine Sorte?
 - Menge von Konstanten
 - Bsp.: Integer: Menge aller positiven und negativen ganzen Zahlen inklusive der Null
- Was ist eine Variable?
 - Platzhalter
 - Kann Werte einer bestimmten Sorte annehmen

Deklaration von Sorten und Variablen

- Sorten: Angabe als Menge von Konstanten

$$S_1 = \{1, 2, 4, 7\}$$

$$S_2 = \{i \mid i \in S_1 \text{ und } i < 5\}$$

$$S_3 = S_1 \cap S_2$$

- Variablen: Angabe der Sorte
 x : variable over S_1

Funktionen

- Was ist eine Funktion?
 - Hat eine Signatur

$$f : X_1 \times \dots \times X_n \rightarrow Y_1$$

- Ein Tupel aus dem Definitionsbereich als Eingabe
- Ein Wert aus dem Wertebereich als Ausgabe

Deklaration von Funktionen

- Angabe der Funktion oder des Funktionssymbols und seine Bedeutung
- Zur Ausführung des Petrinetzes muss eine Definition vorliegen

Beispieldeklaration für DP (3P)

$$P = \{p_1, p_2, p_3\}$$

$$G = \{g_1, g_2, g_3\}$$

$$l, r : P \rightarrow G$$

x : variable over P

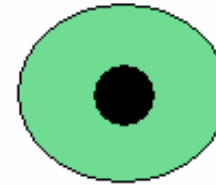
$$l(p_1) = r(p_2) = g_1$$

$$l(p_2) = r(p_3) = g_2$$

$$l(p_3) = r(p_1) = g_3$$

Plätze in High-Level-PN

- Vorher: Platz
 - markiert mit einer oder mehreren Marken
 - Marken nicht unterscheidbar
- Jetzt: Platz
 - Markiert mit einer oder mehreren Marken
 - Marken repräsentieren Konstanten
 - Marken sind Datenträger
 - Marken sind unterscheidbar



Konsumieren

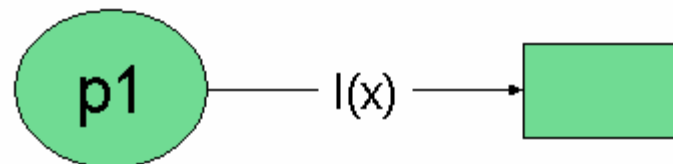
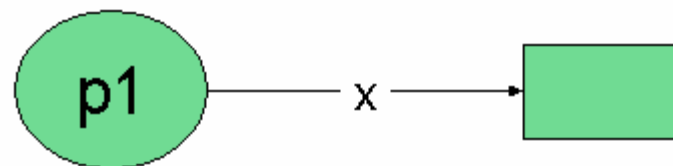
- Vorher:

- Leer oder
- Bei gewichteten Kanten: *Anzahl* der Token



- Jetzt:

- Angabe, *welche* und *wie viele* Marken konsumiert werden
- Term aus Konstanten, Variablen und Funktionen

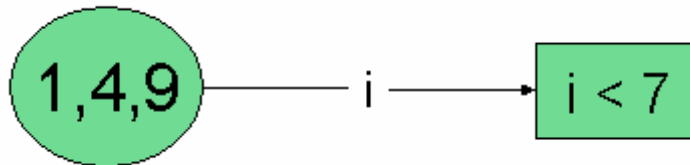


Neue Schaltbedingung

- Wann darf eine Transition schalten?
- Definition Modus:
 - Ein Modus ist
 - eine Belegung aller Variablen
 - eines bestimmten Terms
 - mit den Konstanten aus dem Universum
- Schaltbedingung (vorerst):
 - Falls für jede eingehende Kante gilt, dass ein Modus existiert UND
 - In den zugehörigen Plätzen genug passende Konstanten vorhanden sind

Beschriftung von Transitionen

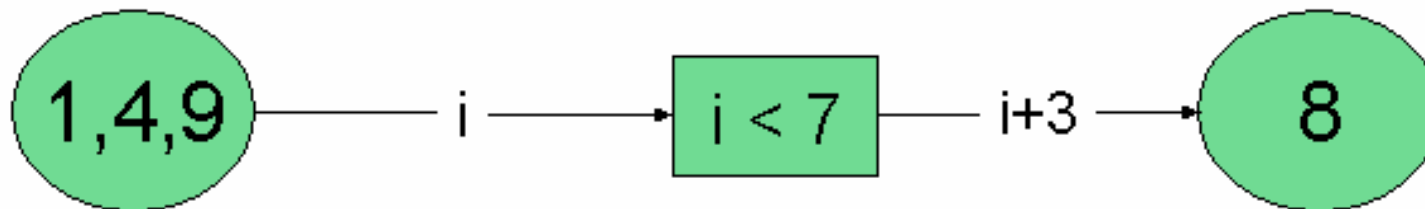
- Guards
 - „Schützen“ die Transition vor ungewollter Ausführung
 - Ausdrücke, die zu „wahr“ oder „falsch“ evaluierbar sind
 - Shortcut Notation für if Anweisungen



- Erweiterung der Schaltbedingung
 - Alle Guards müssen zu „wahr“ evaluieren, damit die Transition schalten kann

Produktion von Marken

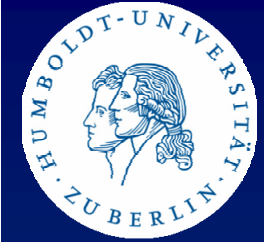
- Beschriftung analog zu Platz \rightarrow Transition
- Produktion von Marken
 - Evaluieren der Funktionen in den Termen
 - Produktion einer Marke, die das Ergebnis repräsentiert
 - Legen der Marke in den Folgeplatz



Zusammenfassung Schaltvorgang

- Finde eine passende Belegung für die Variablen in den Beschriftungen der eingehenden Kanten
- Überprüfe die Guards in der Beschriftung der Transition
- Konsumiere die Marken

- Evaluiere ausgehende Kantenbeschriftungen
- Produziere Ergebnis-Marken

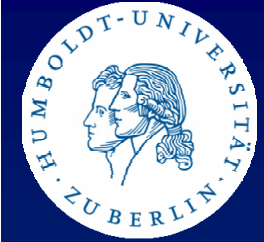


Einige kleine Beispiele

13.02.2008

Ausblick: Was kommt in 14 Tagen?

- Strukturen und Terme
 - Formale Definition der High Level Netze
- Erreichbarkeit in High Level Netzen
- System Schemata



Vielen Dank
für Eure Aufmerksamkeit

13.02.2008