5.2. Entfaltung von CPN

Was: CPN-Modell -> verhaltensgleiches Pl-T-Modell
Wozu? Manche Eigenschaften sind dort besser analysierbar.
Simulation evtl. effektiver.
Implementierung evtl. einfacher.

Erläuterung mit ziemlich elementaren Strukturen:

- 2 Farben
- V=1
- Max. 2 VB und 2 NB

PN47:
Oben links CPN mit oder , rechts entfaltet

Plätze: p mit Kapazität für zwei Farben erzeugt zwei Plätze, jeder logisch einer Farbe zugeordnet

Transitionen: oder zwischen 2 Farben erzeut zwei Transitionen mit einer Kante von dem der entsprechenden Farbe zugeordneten Platz

PN47:
Unten links CPN mit und, rechtsentfaltet

Plätze w.o.
Trans. mit und zw. 2 Farben erzeugt eine Trans. mit je einer Kante von dem der entsprechenden Farbe zugeordneten Platz (eFzP) zu dieser einen Transition

PN48:
w.o., je nur eine Farbe in der Aufzählung:

genau eine Kante von dem eFzP

w.o.zwei Farben in der Aufzählungen
genau zwei Kanten zu den der eFzP

PN49:
w.o. Farbvariable in VB, NB gleich, beide Farben:
2 Transitionen (oder) von den der eFzP zu eFzP

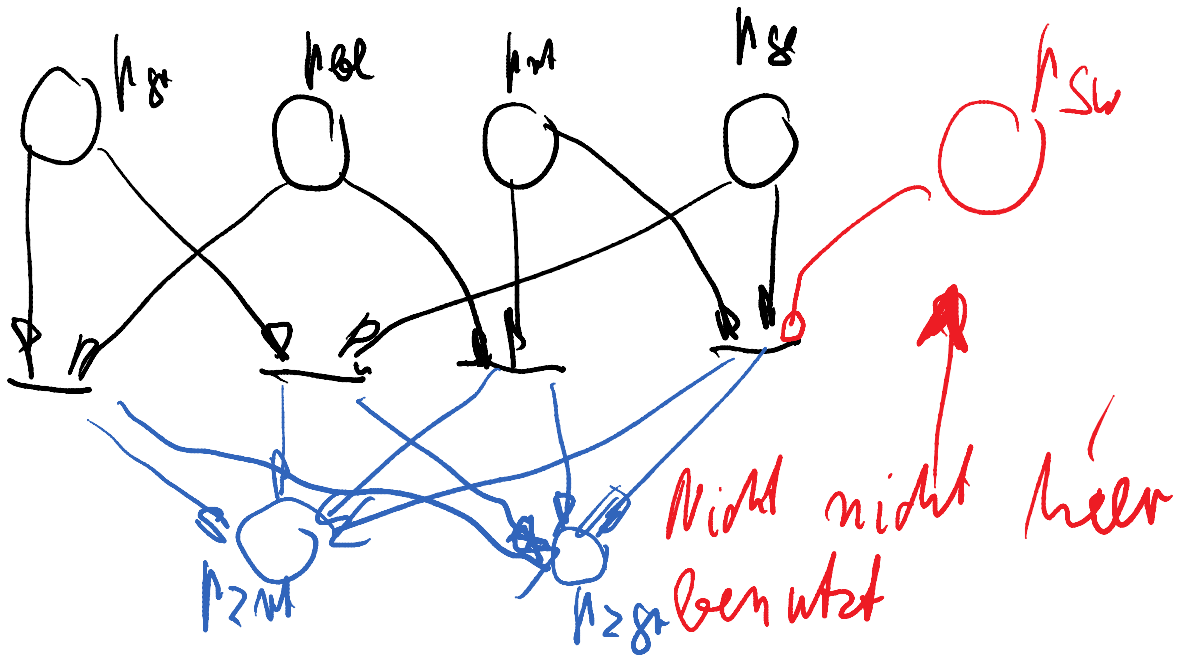
w.o. Farbvariable in VB, NB getauscht:
2 Transitionen (oder) von den der eFzP zu dem jeweils anderen der FzP

Verallgemeinern bei klomlexen Booleschen Ausdrücken.

z.B.: $(gr \vee ma) \wedge (bl \vee ge)$

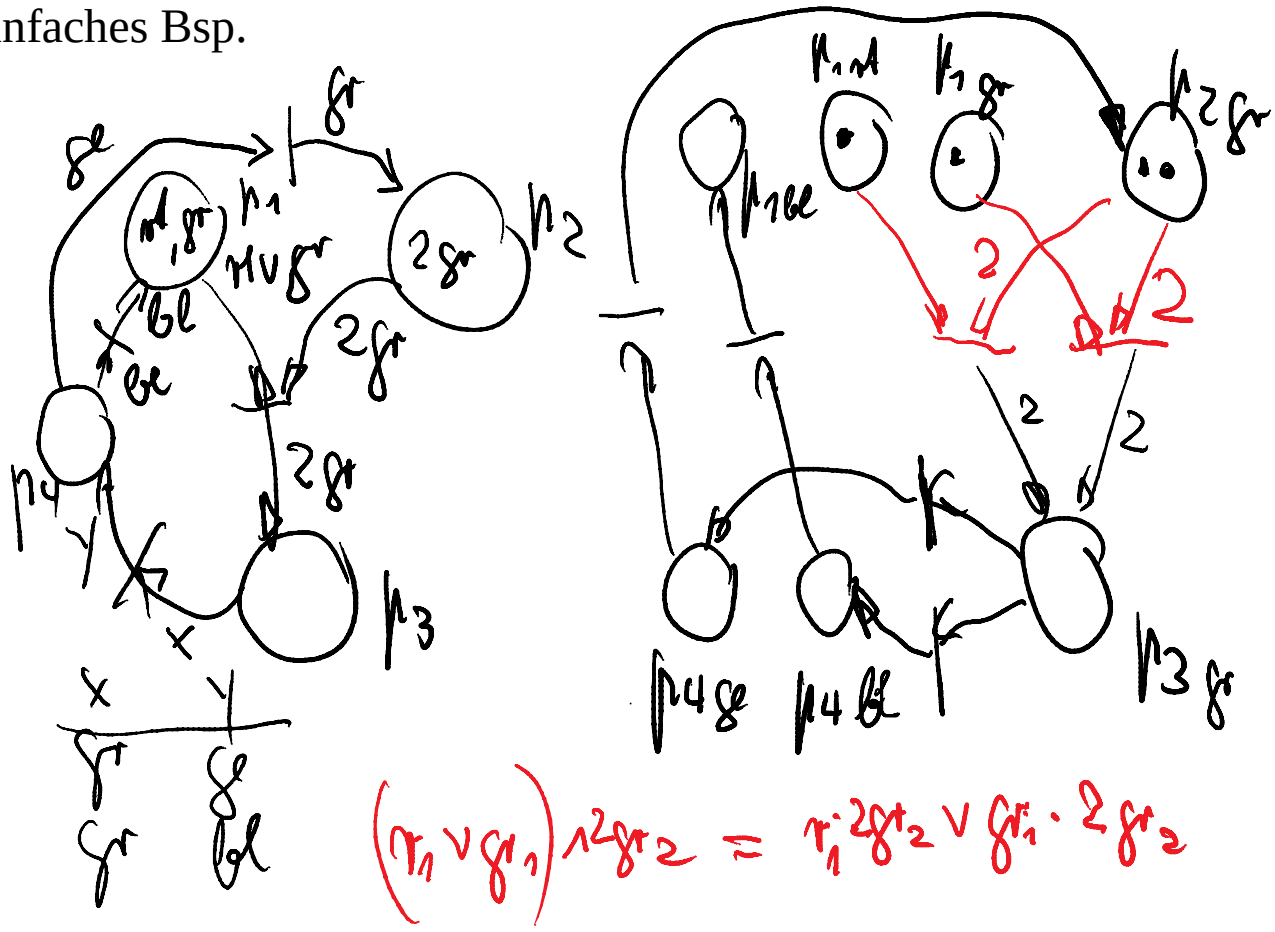
disjunktive Normalform:

$= gr \cdot bl \vee gr \cdot ge \vee ma \cdot bl \vee ma \cdot ge \cdot \overline{sw}$



Nachbed. nur Aufzählung. rt, gr

Einfaches Bsp.



Inschub über Auswahl 6. 7. 8. 9.

Mehrheit der anwedenen nicht enthaltenen Stimmen votiert für 6.

7. 8. 9. Sind in der Prüfungsperiode WS 2012/13 kein Gegenstand der mündlichen Prüfung !!!! Den Prüfer bitte daran erinnern, fals nicht mehr transparent.

6. Modellierung paralleler und verteilter Programme mit PN

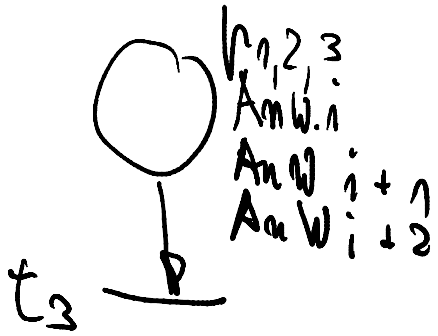
Wozu? Analyse von PN-Eigenschaften auf reale Programme (Steuerstruktur)

Auszug aus den Grundstrukturen:

6.1. Sequentielle Strukturen

Im Weiteren Softwareaktionen werden den Plätzen zugeordnet (wap): Wenn ein p wap neu markiert wir, wird die zugeortnete wap gestartet. Folgetransitionen von p sind im Normalfall nur sf, wenn die Aktion beendet ist.

PN53 rechts, rein sequntiell, nur sinnvoll, wenn an t1,t2 Synchronisationskanten von und oder zu anederen parallelen Prozessen verbunden sind.
Sonst:



Alle verzweigenden Strukturen: wx an Konfliktransitionen, wx wird aus der Datenstruktur abgeleitet.

Bsp. Rekursion (PN54 rechts) für sinnvolle Anwendungen der CPN auf diesem Gebiet.
Idee Rekursionstiefe durch Farben, eine Ebene tiefer ci -> ci+1;
Eine Ebene höher ci+1 -> ci, bei c0 verlassen der rekursiven Prozedur.

6.2. Prozessmodelle

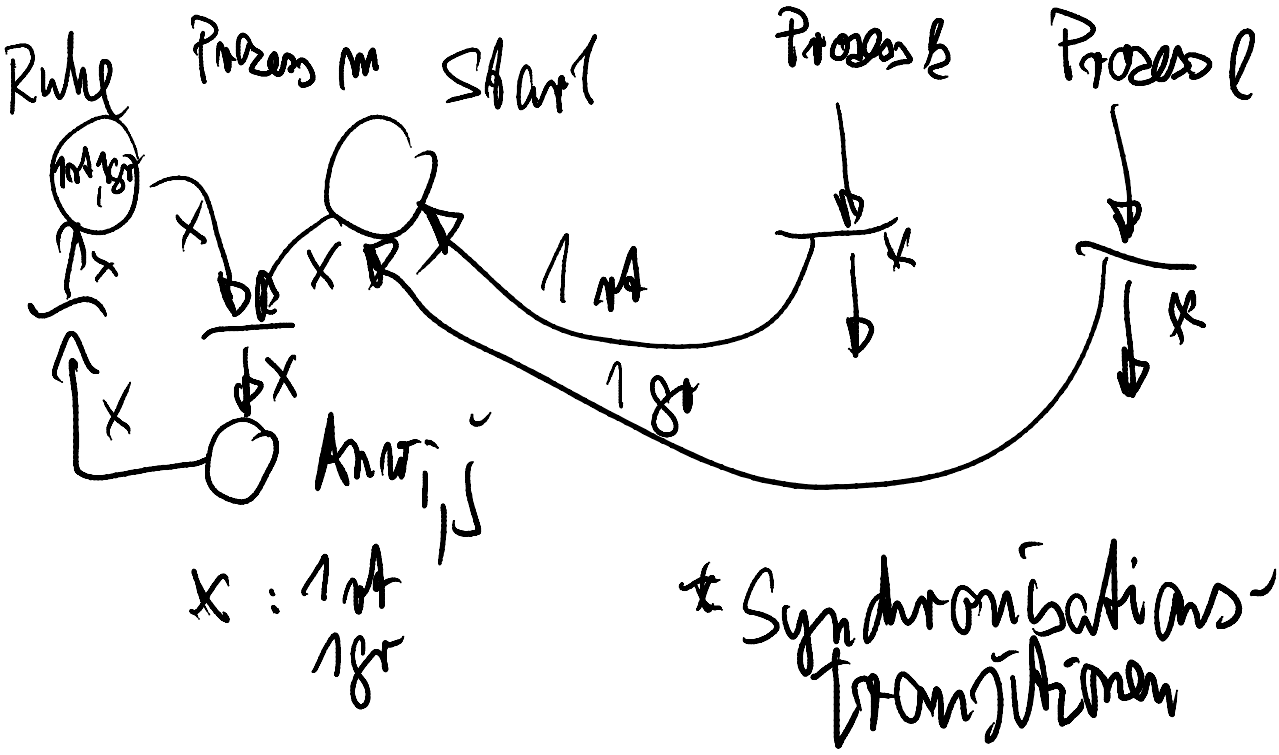
Paralle oder quasiparallele zyklische Teilprogramme (nicht zyklisch als Ausnahme)

PN 55 links
Grundform, nicht wiedereintrittsfähig durch die eine Marke in Ruhe.

Start: markierung kennzeichnet, dass die Anweisungen (Bsp. hier Anwi, Anwj) sequentiell bearbeitet werden sollen)

Ruhe: nur mit marke kann der Prozess neu gestrtet werden.

Modifikation für wiedereintrittsfähig (Bsp.aufrufbar durch zwei Prozesse)



Erweiterungen zum Grundmodell (nicht prüfungsrelevant):

- Erweiterungen modellieren die üblichen Prozessmöglichkeiten in Echtzeitbetriebssystemen
- PN 55links, PN56 … PN58

6.3. Synchronisation zwischen Prozessen

(prüfungsrelevant nur die hier behandelten, diese sind typisch)

6.3.1. Folge seuqentiell streng
PN 60

- ➔ Anwi und Anwj nicht geichzeitig, einzige mögliche Folge: Anwi, Anwj, Anwi, Anwj, …

6.3.2. Exklusiver Ausschluss
PN 64

- ➔ Anwi, Anwj nicht gleichzeitig, Folge beliebig
- ➔ Praktische Anwendung Zwei Nutzer ein Drucker (PN 15)
- ➔ Konflikt zwischen t1 und t3, muss gelöst werden

6.3.3. Rendezvous
PN 66

- ➔ Anwi, Anwj immer geichzeitig gestartet (und, in PN66 aber nicht realisiert, gleichzeitig beendet)
- ➔ Warten eines Prozesses mit Marke in WaitStartk oder j
- ➔ Bei Marke in beiden Plätzen Start von Anwi und j durch transition Syncstart
- ➔ Gleichzeitiges Ende -> Zusammenfassen von t2 und t4

6.4. Kommunikation zwischen Prozessen

(prüfungsrelevant nur die hier behandelten, diese sind typisch)

6.4.1. Erzeuger-Verbraucher-Kommunikation
PN 68

- ➔ Kommunikation zwischen zwei Prozessen über Puffserspeicher
- ➔ Reihenfolge Puffer schreiben, Puffer leeren, Puffer schreiben, Puffer leeren, …, nicht gleichzeitig Schreiben und lesen
- ➔ Entspricht in der Synchronisationsstruktur „Strenge sequentielle Folge)

6.4.2. Mailbox (Briefkasten)
Bsp. PN71

- ➔ Gefärbtes Netz
- ➔ Mehrere Puffer
- ➔ Mehrere Prozesse
- ➔ Adressierung durch Farben

Bsp:
2 Nachrichtentypen (Zieladressen) ni, nj
2 Prozesse schreiben ni
1 Prozess schreiben nj
1 Prozess liest ni
2 Prozesse lesen nj

Kommunikation läuf so:

1. Es ex. mind. ein freier Puffer (m(PufferFrei)>=1)
2. Schreibprozess generiert ni oder nj (je nach Prozess) in einem Puffern in Nachrichten und kennzeichnet das durch eine Marke vom Typ ni oder nj im PlATZ Nachrichten und erzeut einen normale Marke in Platz PufferVoll und entzielt eine Marke in PufferFrei
3. EinLeseprozess entnimmt ni bzw nj (je nach Prozess und vorhandener Nachricht), entzieht dem Platz Nachrichten eine Marke vom Typ ni oder nj (je nach Prozess) und erzeugt eine normale Marke in Puffer leer und entzieht eine Marke von Typ ni bzw. nj von Nachrichten und entzieht in Platz Puffer voll eine normale Marke.

4. Mehrere Kommunikationen sind je nach Nachrichten und freien Puffern parallel (gleichzeitig) möglich.