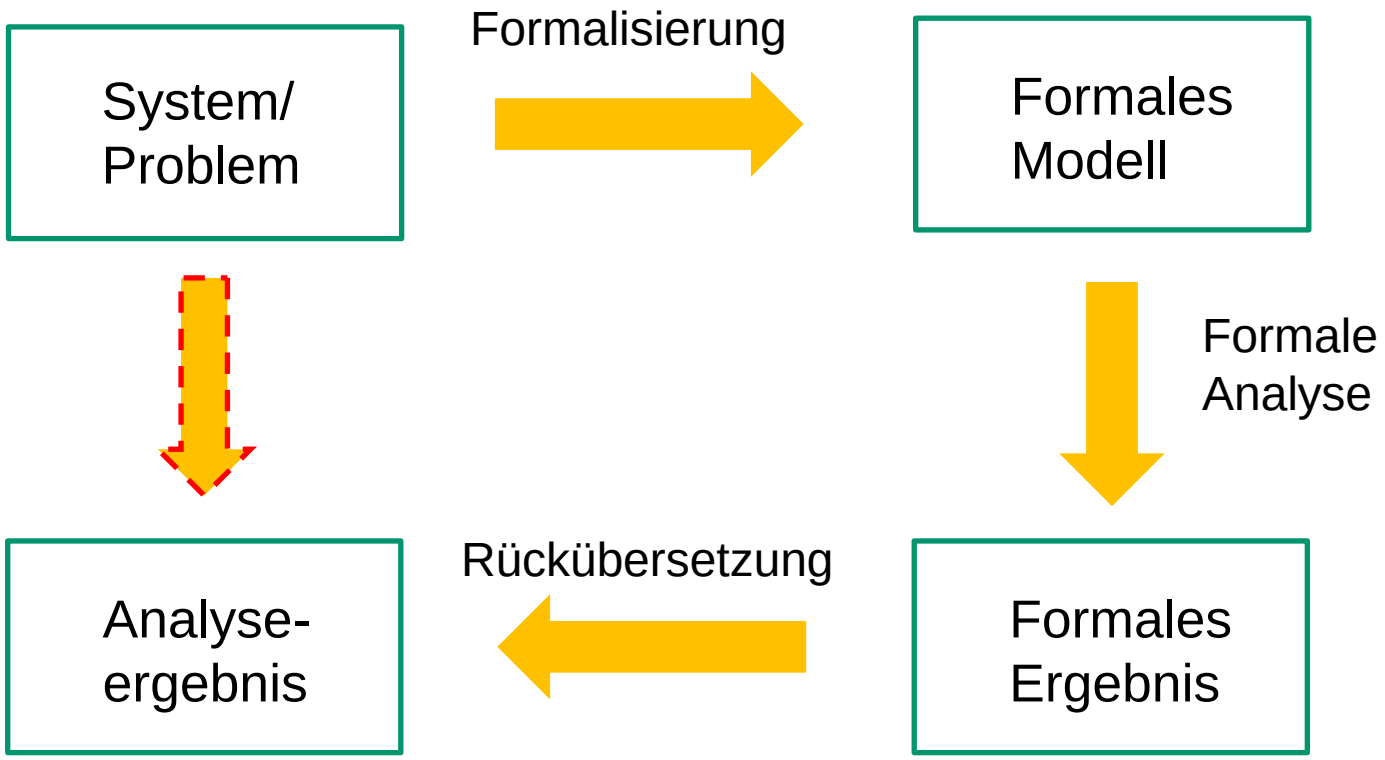
Zusammenfassung und Ausblick

Jens Kosiol

15. Juli 2024

Integration von formalen Methoden: Genereller Ansatz



Ansatz der Vorlesung

- Formales Model: (getypte, attributierte) Graphen und Graphtransformationssysteme zur Modellierung dynamischen Verhaltens
- Formale Analysemethoden auf diesem Modell
  - *Geschachtelte Graphformeln und Generierung von Anwendungsbedingungen*
  - *Konflikt- und Abhängigkeitsanalyse*
  - *(Terminations- und Konfluenzanalyse)*
  - *(Model Checking von Graphtransitionssystemen)*
- Anwendung auf
  - *Programmieren mit Graphgrammatiken*
  - *Spezifizieren und Suchen von Services*
  - *Extraktion von Visual Contracts*
  - *Entwurf von domänenspezifischen Modellierungssprachen*
  - *Erkennen inkonsistenter Anforderungen*
  - *Modellbasiertes Testen*

Modellierung von Datenstrukturen mit Graphen

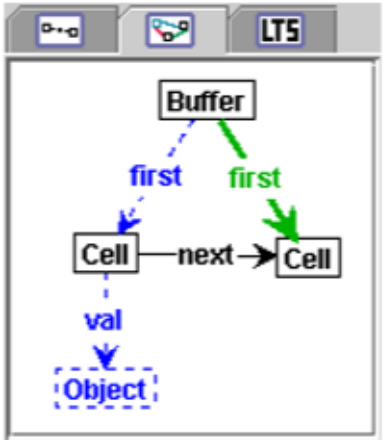
- Mit Graphen kann man verschiedenste Datenstrukturen modellieren.
  - *Listen, Stacks, Bäume, etc.*
- Graphknoten
  - *Markierungen (Typisierung) können verschiedene Arten von Knoten unterscheiden.*
  - *Markierungen können auch weitere Informationen halten (Attributierung).*
- Graphkanten
  - *Kanten können Relationen zwischen Knoten spezifizieren.*
  - *Kantenmarkierungen können verschiedene Arten von Kanten unterscheiden.*

Graphtransformation

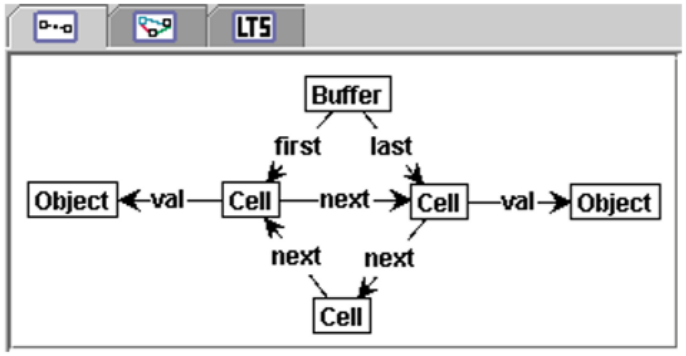
- Änderungen von Graphen werden regelbasiert spezifiziert.
- Eine Graphtransformationsregel
  - *beschreibt alle Änderungen (Löschungen und Hinzufügungen), die bei ihrer Anwendung durchgeführt werden, explizit;*
  - *spezifiziert ggf. zusätzlichen Kontext, bei dessen Vorhandensein die Regel nicht angewendet wird (NACs bzw. allgemeine Anwendungsbedingungen).*
- Graphprogramme (Regelanwendung, sequentielle Komposition und Iteration) bilden ein vollständiges Berechnungsmodell auf Graphen.

Beispiel: Graphtransformation

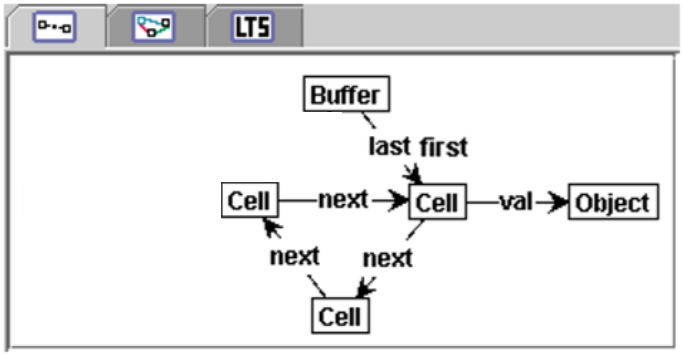
Regel get:



Graph G:



Graph H:

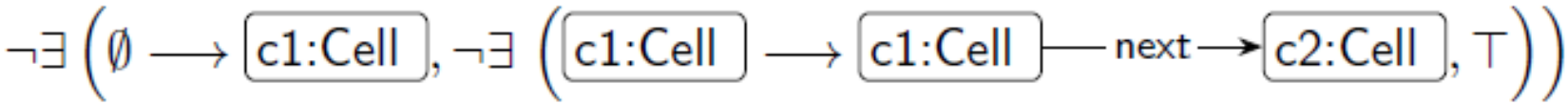


Geschachtelte Graphbedingungen

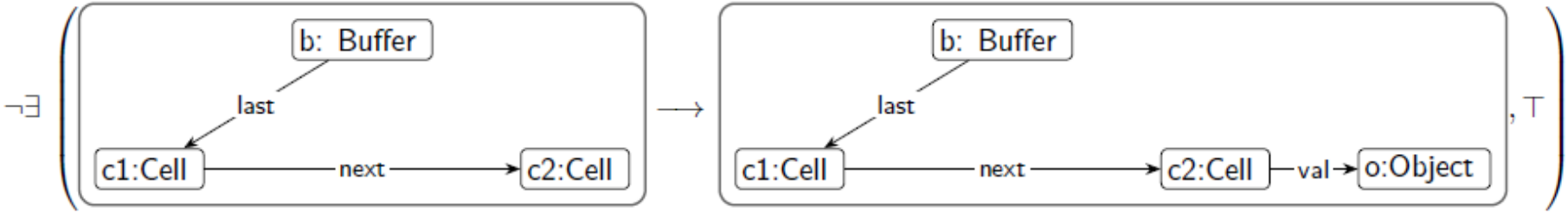
- Eigenschaften von Graphen können als sog. geschachtelte Graphbedingungen ausgedrückt werden.
  - *Fordern das (Nicht-)Vorhandensein von Untergraphen, für die dann weitere Eigenschaften gelten*
  - *Bieten eine Logik 1. Ordnung auf Graphen*
  - *Kreisfreiheit z.B. nicht ausdrückbar*
- Geschachtelte Graphbedingungen können so als Anwendungsbedingungen in Regeln integriert werden, dass keine Regelanwendungen mehr möglich sind, die ungültige Graphen als Resultat haben.

Beispiele geschachtelte Graphformeln

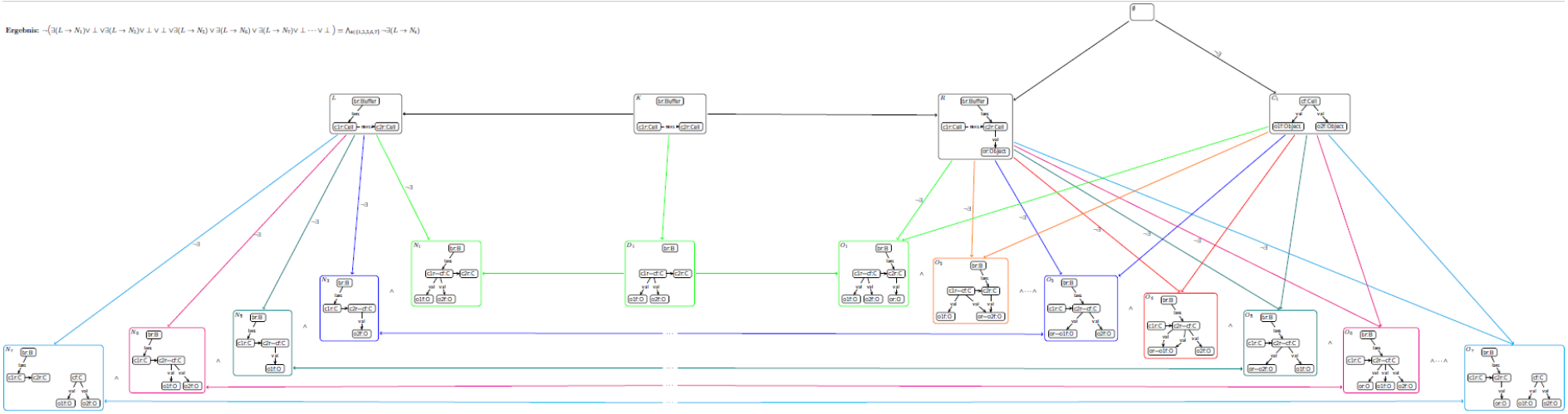
- Für jede Zelle gilt, dass eine next-Kante von ihr ausgeht.



- Ein Morphismus aus dem ersten Graphen (LHS der *put*-Regel) soll Zelle c2 so abbilden, dass diese kein Object besitzt.



Berechnung von garantierenden Anwendungsbedingungen



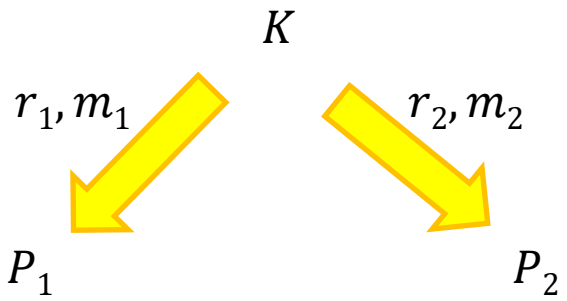
Definition: Parallel unabhängige Regelanwendungen

Gegeben zwei Regeln $r_1 = (L_1 \supseteq K_1 \subseteq R_1)$ und $r_2 = (L_2 \supseteq K_2 \subseteq R_2)$, dann sind zwei Transformationsschritte $t_1 : G \Longrightarrow_{r_1,m_1} H_1$ und $t_2 : G \Longrightarrow_{r_2,m_2} H_2$ parallel unabhängig, falls sich ihre Regelansätze nur in zu erhaltenden Graphelementen von r_1 und r_2 überschneiden, also gilt:

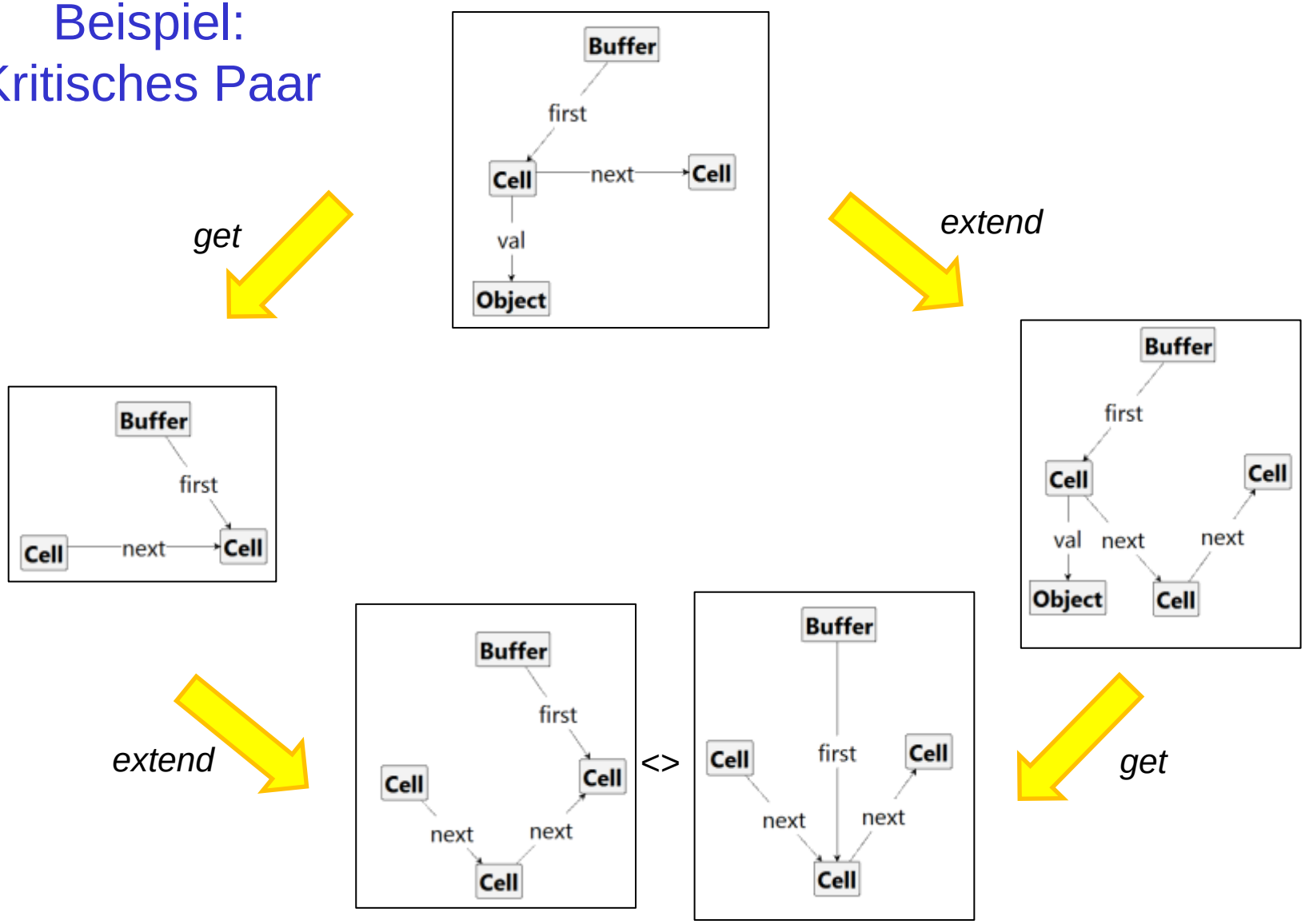
$$m_1(L_1) \cap m_2(L_2) \subseteq m_1(K_1) \cap m_2(K_2).$$

Definition: Kritisches Paar

Gegeben zwei Regeln $r_1 = (L_1, R_1)$ und $r_2 = (L_2, R_2)$, dann ist ein kritisches Paar ein Paar von parallel abhängigen Regelanwendungen $K \Longrightarrow_{r_1,m_1} P_1$ und $K \Longrightarrow_{r_2,m_2} P_2$, sodass $m_1(L_1) \cup m_2(L_2) = K$ gilt.



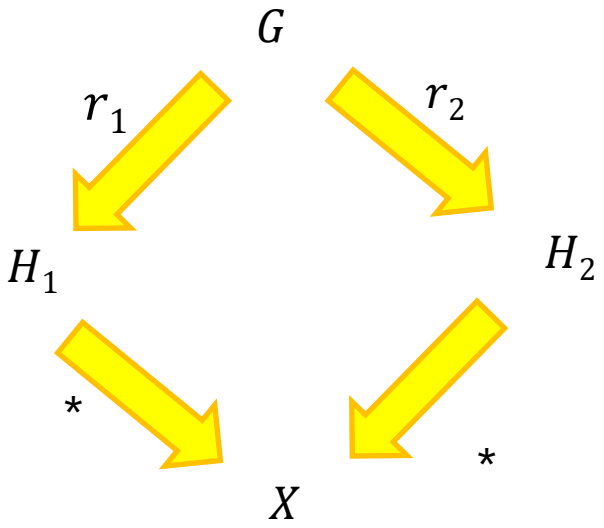
Beispiel: Kritisches Paar



Es gibt keine Regelanwendungen, die das kritische Paar wieder zusammenführen.

Konfluenz: Hinreichende Kriterien

- Theorem: Wenn alle Regeln eines GTS parallel unabhängig sind, dann ist das GTS konfluent.
  - *Beweis: [EEPT06]*
- Wenn es auch parallel abhängige Regelpaare im GTS gibt:
  - *Wenn alle kritischen Paare strikt konfluent sind, dann sind alle Regelanwendungen des GTS lokal konfluent.*
  - *Wenn alle Regelanwendungen des GTS lokal konfluent sind und das GTS terminiert, dann ist das GTS konfluent.*



Anwendungen von Graphtransformation

- Spezifizieren und Suchen von Services (Systementwurf)
  - *(könnte Abhängigkeitsanalyse nutzen)*
- Extraktion von Visual Contracts aus Java-Programmen (Reverse Engineering)
  - *(könnte Model Checking und Generierung von Anwendungsbedingungen nutzen)*
- Entwurf von domänenspezifischen Modellierungssprachen (Modellbasiertes Softwareengineering)
  - *Setzt logischen Formalismus (z.B. geschachtelte Graphbedingungen) voraus*
- Erkennen inkonsistenter Anforderungen (Anforderungsspezifikation)
  - *Nutzt Konflikt- und Abhängigkeitsanalyse*
- Modellbasiertes Testen (Softwarequalität)
  - *Nutzt Konflikt- und Abhängigkeitsanalyse*
- (Programmieren mit Graphgrammatiken)
  - *(kann alle vorgestellten formalen Analysemethoden nutzen)*
- (Übersetzen zwischen Modellen)
  - *Kann Terminations- und Konfluenzanalyse nutzen*

Weitere Anwendungen

- Synchronisation verschiedener Sichten auf ein System (Modellsynchronisation)
- Refaktorierung und Versionsverwaltung von Modellen
- Stochastische Analyse von dynamischen Systemen
- Optimierungsverfahren auf Graphen/Modellen als Datenstrukturen
- Modellierung und Analyse (bio-)chemischer Prozesse mittels Graphtransformation
- Didaktisches, visuelles Modell mit präziser Bedeutung
- …