

# Ressourcenplanung unter Nutzung der Java-Constraint-Bibliothek firstCS

Saskia Sandow

Fraunhofer Institut für  
Rechnerarchitektur und  
Softwaretechnik

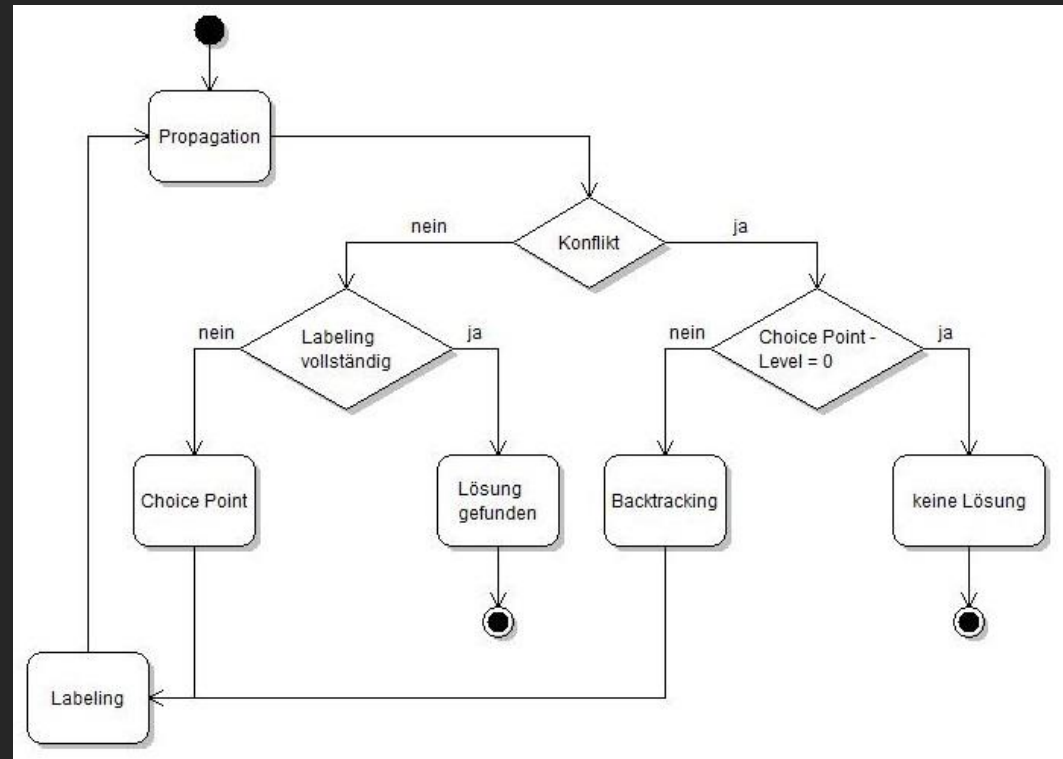


# Überblick

- Ressourcenplanung auf Basis constraint-logischer Programmierung
- Implementierung in Java unter Nutzung der Java-Constraint-Bibliothek firstCS (Wolf 2006)
- Problemspezifikation in XML
- Konfigurierbarkeit über zusätzliche XML-Datei
- Interaktive Nutzung über Benutzeroberfläche

# Constraint-logische Programmierung (CLP)

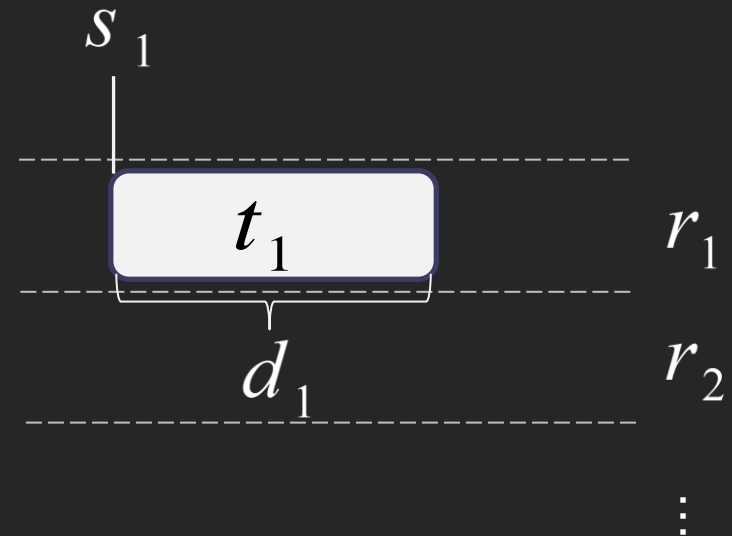
- Problemmodellierung durch Variablen und Constraints (Randbedingungen)
- Definition von Constraints über eine, zwei oder mehr Variablen (entspricht prädikatenlogischer Formel)
- Lösung entspricht einer Wertebelegung aller Variablen unter Einhaltung aller Constraints



➤ Verbesserung der Suche durch Lernen oder Heuristiken zur Variablen- und Wertauswahl

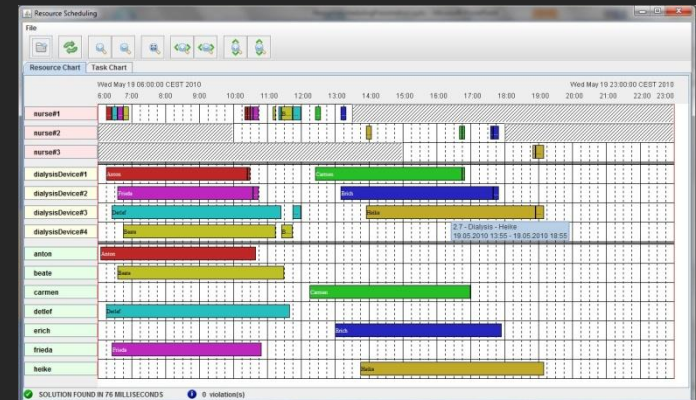
# Ressourcenplanung

- Task (Aufgabe, Arbeitsgang) mit Start, Dauer, Ende und Ressource
- Constraints, z.B. Task  $t_1$  vor Task  $t_2$
- Ressourcen
  - exklusiv
  - alternativ exklusiv
  - kumulativ
  - alternativ kumulativ
- Ressourcenplanungsproblem = zeitliche Festlegung verschiedener Tasks ohne gegenseitige Überlappung (exklusiv) bzw. ohne Überschreitung der Ressourcenkapazität (kumulativ)



# Ressourcenplaner - Überblick

- Problemspezifikation in XML
  - Beispiel Dialysebehandlung
- Planung
  - Einlesen und Verarbeitung der Spezifikation
  - Suche unter Nutzung einer optionalen Konfigurationsdatei
- Benutzeroberfläche zur interaktiven Nutzung
  - graphische Darstellung eines Plans
  - Änderungen am Plan durch den Benutzer

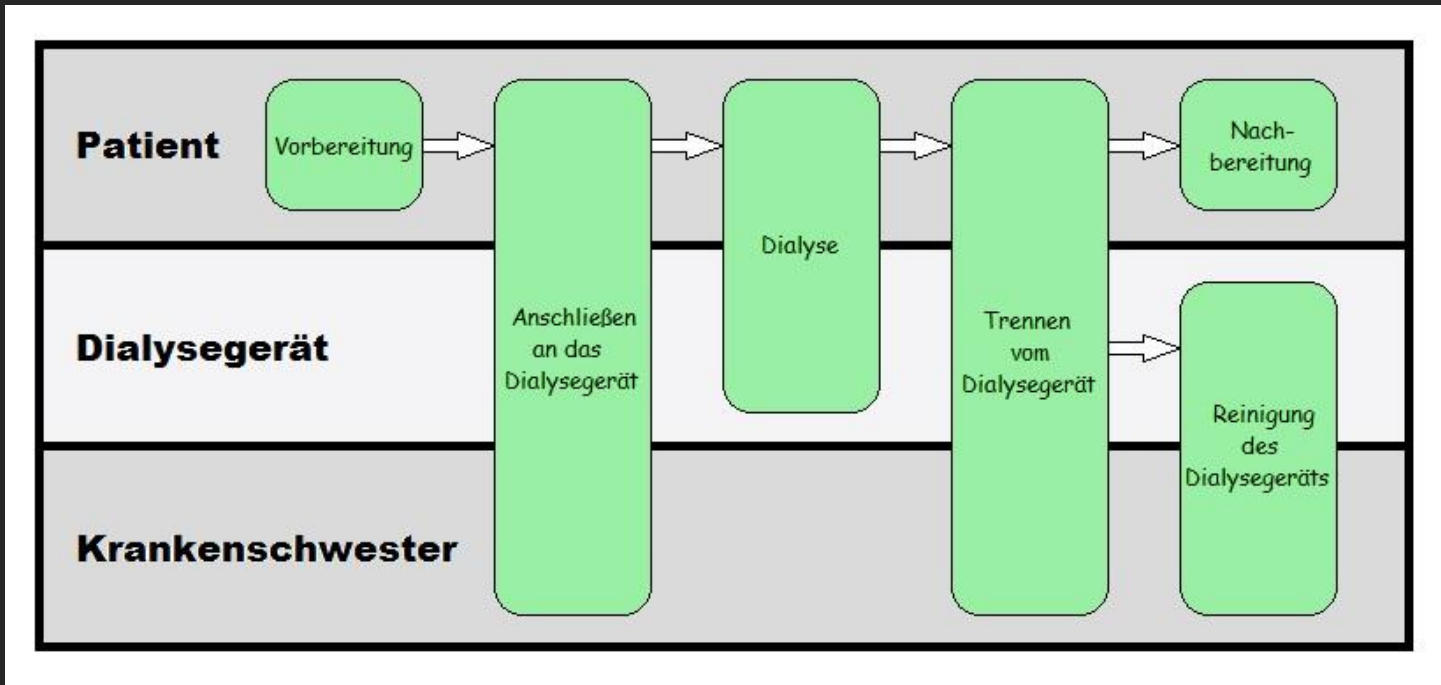


# Problemspezifikation

- Erforderliche XML-Elemente :
  - `makeSpan` : Planungshorizont
  - `tasks` : Tasks mit erforderlichen und optionalen Attributen sowie einer Auflistung potentieller Ressourcen
  - `relations` : Constraints, z.B. zur Festlegung von Reihenfolgen
  
- Optionales XML-Element :
  - `resourceGroups` : vom Benutzer festgelegte Gruppierungen von Ressourcen
  
- Validierung der Spezifikation durch XML-Schema

# Problemspezifikation

## Beispiel Dialysebehandlung



- Parsen der Spezifikation und Erstellung des Constraint-Systems mit Hilfe der Java-Constraint-Bibliothek firstCS (Wolf 2006)
- Constraints :
  - `AlternativeResource` für jede Ressourcenklasse
  - Element für Tasks, deren Dauer von der Ressource abhängt
  - `AllEqual`, `AllDifferent` und `Before` für die in relations spezifizierten Constraints
  - `IsInDomain` und `WeightedSum` zur Einhaltung von Wunschzeiten (falls angegeben)

# Planung - Suche

- Eingebaute Suche durch einfaches Backtracking
- Ziel = Nutzung der Konfigurierbarkeit über zusätzliche XML-Datei zur einfachen und schnell anpassbaren Steuerung der Suche
- firstCS :
  - Vordefinierte Labeler
  - Erweiterung um eigene Labeler
  - Kombination von Labeler durch MetaLabel
- Sicherstellung des Labelings aller Variablen (direkt oder indirekt)

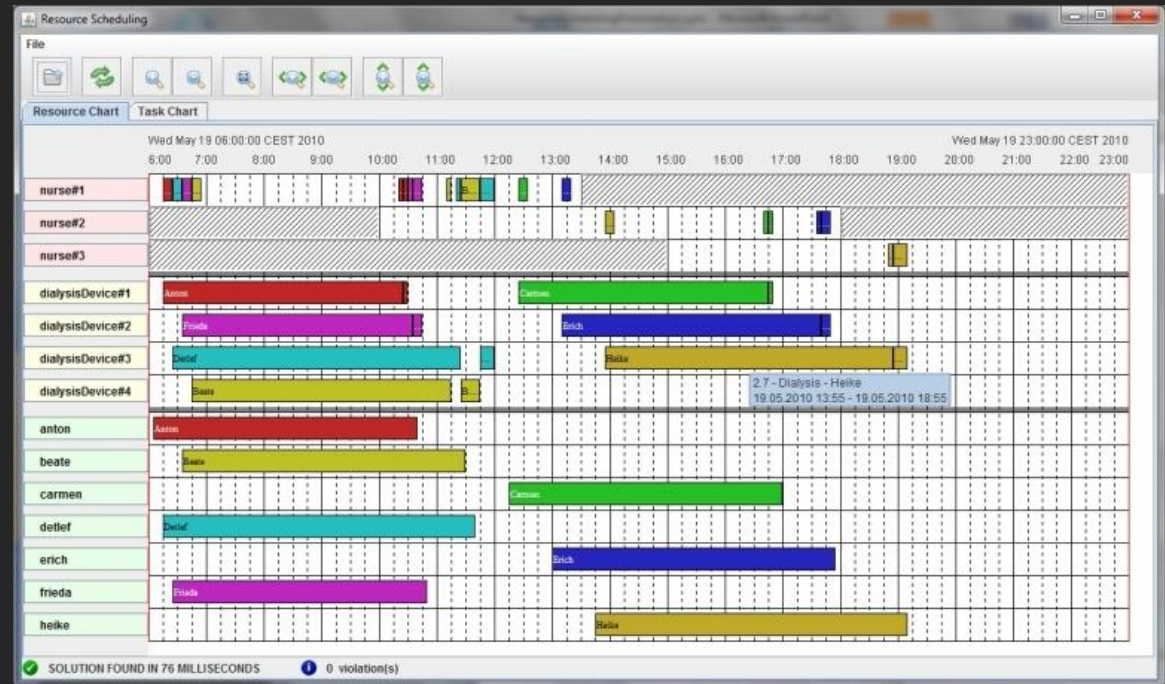
```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <resourceSchedulingRequestSearchStrategy>
3   <search ref="MetaLabel">
4     <search ref="MetaLabel">
5       <forEachGroup groupOrder="id">
6         <search ref="SingleTaskOnAlternativesScheduler">
7           <task taskID="2.*"/>
8           <int val="4"/>
9         </search>
10        <search ref="BtLabel">
11          <task taskID="3.*" var="resource"/>
12          <task taskID="4.*" var="resource"/>
13          <task taskID="5.*_b" var="resource"/>
14        </search>
15      </forEachGroup>
16    </search>
17    <search ref="BtLabel">
18      <forEachGroup>
19        <task taskID="1.*" var="start"/>
20      </forEachGroup>
21    </search>
22    <search ref="BtLabel">
23      <forEachGroup>
24        <task taskID="5.*_a" var="start"/>
25      </forEachGroup>
26    </search>
27  </search>
28 </resourceSchedulingRequestSearchStrategy>
```

# Planung - Optimierung

- Zielfunktion = Einhaltung von Wunschzeiten / Terminen
- Realisiert durch die Constraints zur Einhaltung der Wunschzeiten und einer Variablen für die Anzahl der Verletzungen
- Verletzung tritt ein, falls Wunschzeit + Toleranz nicht erreicht wird
- Suche durch schrittweise Erhöhung der Verletzungen nach Timeout

# Visualisierung

- Graphische Benutzeroberfläche
  - Ressourcenansicht



- Aktivitätenansicht

# Visualisierung - Interaktion

- Interaktive Änderungen durch
  - Verschieben einer Task
  - Deaktivieren einer Ressource
- Umplanung durch so wenig Änderungen wie möglich
- Suchstrategie bleibt erhalten
- Letzter Plan wird gespeichert zur Wiederherstellung beim Finden keiner Lösung

# Benchmarks

	Standardsuche			benutzergesteuerte Suche		
	Timeout			Timeout		
	1000	2000	5000	1000	2000	5000
<b>dialysis7on4</b>	9.3(0)	7.8(0)	5.0(0)	9.2(0)	11.1(0)	11.1(0)
1. 2.7 auf 16:00	12.6(0)	4.6(0)	6.3(0)	7.9(0)	8.0(0)	7.9(0)
2. 2.6 auf 8:00	-(-)	-(-)	-(-)	288.4(1)	287.0(1)	283.9(1)
3. nurse1 gesperrt	-(-)	-(-)	-(-)	2647.5(4)	4632.8(4)	10620.6(4)
<b>dialysis7on4-clean</b>	9.3(0)	6.4(0)	6.3(0)	6.3(0)	7.9(0)	7.8(0)
1. 2.7 auf 16:00	9.5(0)	7.8(0)	9.1(0)	6.2(0)	1.6(0)	4.7(0)
2. 2.6 auf 8:00	-(-)	-(-)	-(-)	81.0(1)	73.6(1)	78.0(1)
3. nurse1 gesperrt	-(-)	-(-)	-(-)	2357.1(4)	2616.6(4)	10344.5(4)
<b>dialysis20on10</b>	2118.2(2)	4111.8(2)	10114.6(2)	3112.7(3)	6119.2(3)	15095.1(3)
1. 2.7 auf 16:00	76.4(0)	74.9(0)	75.0(0)	56.3(0)	62.5(0)	59.4(0)
2. 2.7 auf 8:00	-(-)	-(-)	-(-)	1157.5(1)	2154.3(1)	5154.4(1)
3. nurse4 gesperrt	-(-)	-(-)	-(-)	-(-)	-(-)	-(-)
<b>dialysis20on10-clean</b>	1399.9(2)	2454.4(2)	5363.4(2)	2186.6(3)	4166.1(3)	10158.2(3)
1. 2.7 auf 16:00	38.8(0)	35.9(0)	38.9(0)	39.0(0)	37.5(0)	36.0(0)
2. 2.7 auf 8:00	-(-)	-(-)	-(-)	115.7(1)	115.7(1)	112.3(1)
3. nurse4 gesperrt	-(-)	-(-)	-(-)	1157.4(2)	2162.2(2)	5165.1(2)
<b>surgery</b>	-(-)	-(-)	-(-)	6303.2(7)	12351.4(7)	30299.6(7)
1. 2.4_a auf 16:00				15.6(0)	12.2(0)	15.8(0)
2. 2.6_a auf 8:00				830.4(1)	802.2(1)	812.6(1)
3. nurse1 gesperrt				7559.8(9)	14489.6(9)	35537.2(9)

Intel(R) Xeon(R) E5540 (2,53 GHz) mit 6 GB RAM

Durchschnittszeiten in ms (Anzahl der Verletzungen)

# Benchmarks

- Einfaches Backtracking vs. Benutzergesteuerte Suche über die Konfigurationsdatei
  - Einfaches Backtracking scheitert bei Interaktionen, die Verletzungen hervorrufen
  - Verbesserung des Backtracking z.B. durch zufällige Variablenauswahl
- Suche integriert im Code vs. Einlesen und Verarbeiten der Suche aus der Konfigurationsdatei
  - Kein nennenswerter Laufzeitverlust
- Verschiedene Timeouts
  - Keine Veränderungen an den Lösungen

# Zusammenfassung

- Ressourcenplanung auf alternativen exklusiven Ressourcen
- Problemspezifikation auch für in CLP unerfahrene Benutzer
- Konfiguration der Suchstrategie (Labeling + Variablenauswahl) für firstCS-vertraute Benutzer
- Interaktive Nutzung über die Benutzeroberfläche
  - Schieben einer Task auf eine bestimmte Zeit
  - Deaktivieren einer Ressource

- Erweiterung der Konfigurationsmöglichkeiten
  - Timeout
  - Toleranzspanne für Verletzungen
  - Zielfunktion
- Erweiterung des Ressourcenplaners
  - Kumulative Ressourcen (Erweiterung der Spezifikationsprache)
  - Zusätzliche Constraints in `relations`
- Erweiterung der interaktiven Nutzung
  - Hinzufügen und Entfernen von Tasks oder Task-Gruppen
  - Schieben einer Task auf eine bestimmte Ressource
  - Prioritäten auf mehrere Interaktionen

# Vielen Dank für Ihre Aufmerksamkeit!

