Using Graph Transformation and (M)ILP to solve the IHTC 2024

EURO 2025 Conference – Automated Timetabling Stream

Maximilian Kratz (Real-Time Systems Lab, Technical University of Darmstadt, Germany)

Steffen Zschaler (Department of Informatics, King‘s College London, United Kingdom)

Jens Kosiol (Software Engineering Group, Philipps-Universität Marburg, Germany)

Andy Schürr (Real-Time Systems Lab, Technical University of Darmstadt, Germany)

Jule Pfau (Technical University of Darmstadt, Germany)

Real-Time Systems Lab

Prof. Dr. rer. nat. Andy Schürr

Dept. of Electrical Engineering and Information Technology

Dept. of Computer Science (adjunct Professor)

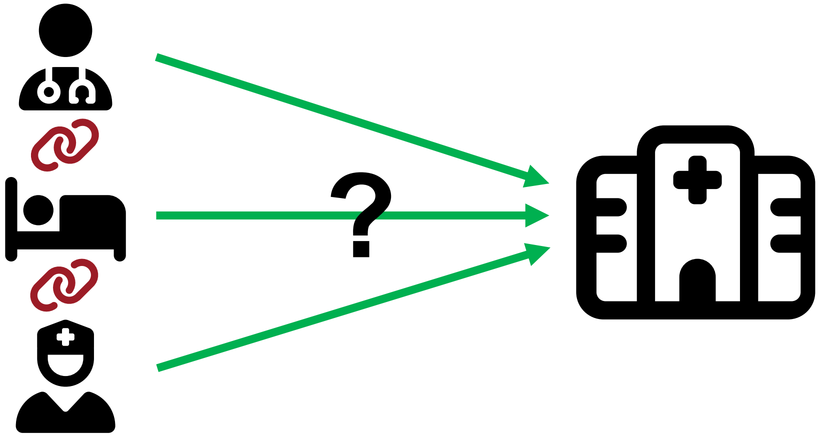
Feel free to contact me: maximilian.kratz@es.tu-darmstadt.de

www.es.tu-darmstadt.de

IHTC 2024 (a quick summary)

- **I**ntegrated **H**ealthcare **T**imetabling **C**ompetition **2024**
- Integrated combinatorial optimisation problem in the hospital context

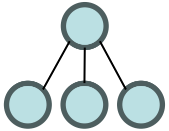
1. Surgical case planning
2. Patient admission scheduling
3. Nurse-to-room assignment



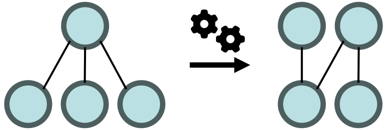
(All icons from https://fontawesome.com)

Our Motivation

- We work on …
  - Model-Driven Software Engineering
  - Models as Graphs



  - Graph Transformation & Optimization of Graphs



We want to solve the challenge using our methods & tools.

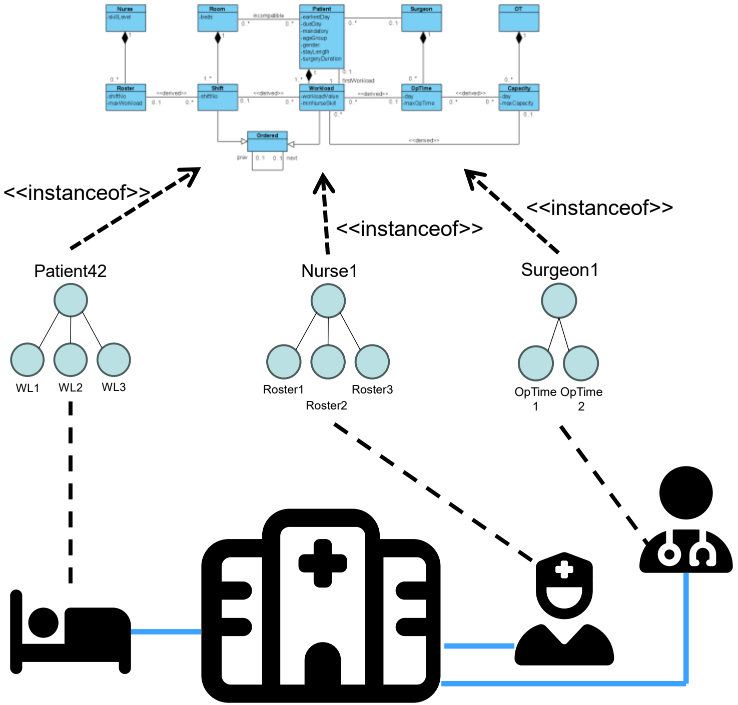
(All icons from https://fontawesome.com)

Layers in Model-Driven Software Engineering

M2: Metamodel !

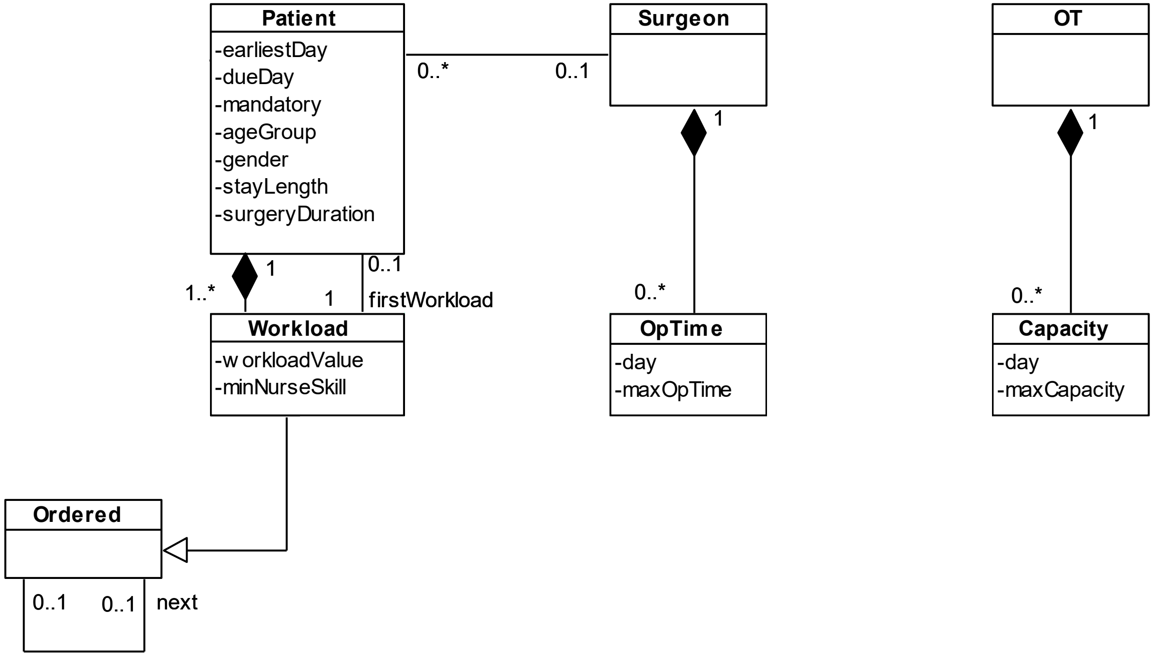
M1: Model

M0: Real-world entities

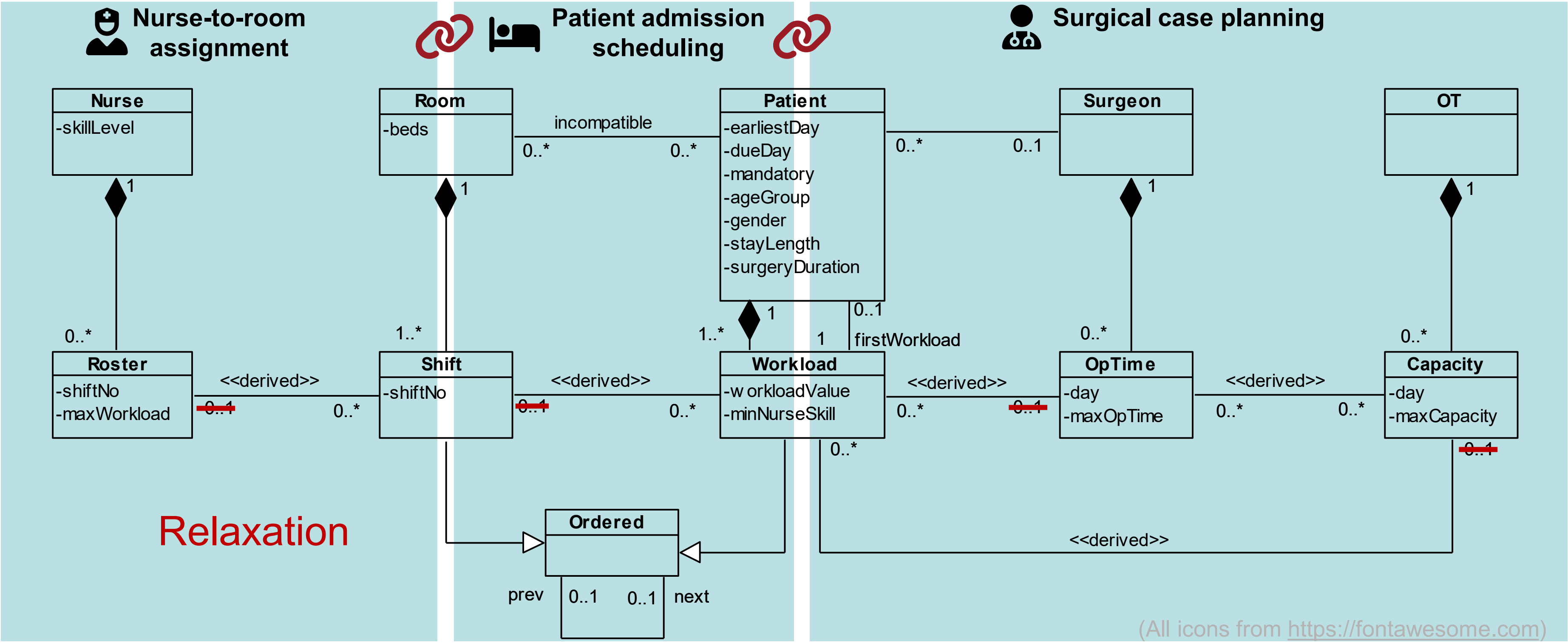


(All icons from https://fontawesome.com)

A (simplified) Metamodel for the IHTC 2024



A (simplified) Metamodel for the IHTC 2024



Nurse-to-room assignment

Patient admission scheduling

Surgical case planning

Nurse
-skillLevel

Room
-beds

Patient
-earliestDay
-dueDay
-mandatory
-ageGroup
-gender
-stayLength
-surgeryDuration

Surgeon

OT

Roster
-shiftNo
-maxWorkload

Shift
-shiftNo

Workload
-workloadValue
-minNurseSkill

OpTime
-day
-maxOpTime

Capacity
-day
-maxCapacity

Ordered

incompatible 0..* 0..*

0..* 0..1

1 0..*

1 1..*

1 1..*

1 0..*

1 0..*

0..1 1 firstWorkload

<<derived>> ~~0..1~~ 0..*

<<derived>> ~~0..1~~ 0..*

<<derived>> 0..* ~~0..1~~

<<derived>> 0..* 0..*

<<derived>> 0..* ~~0..1~~

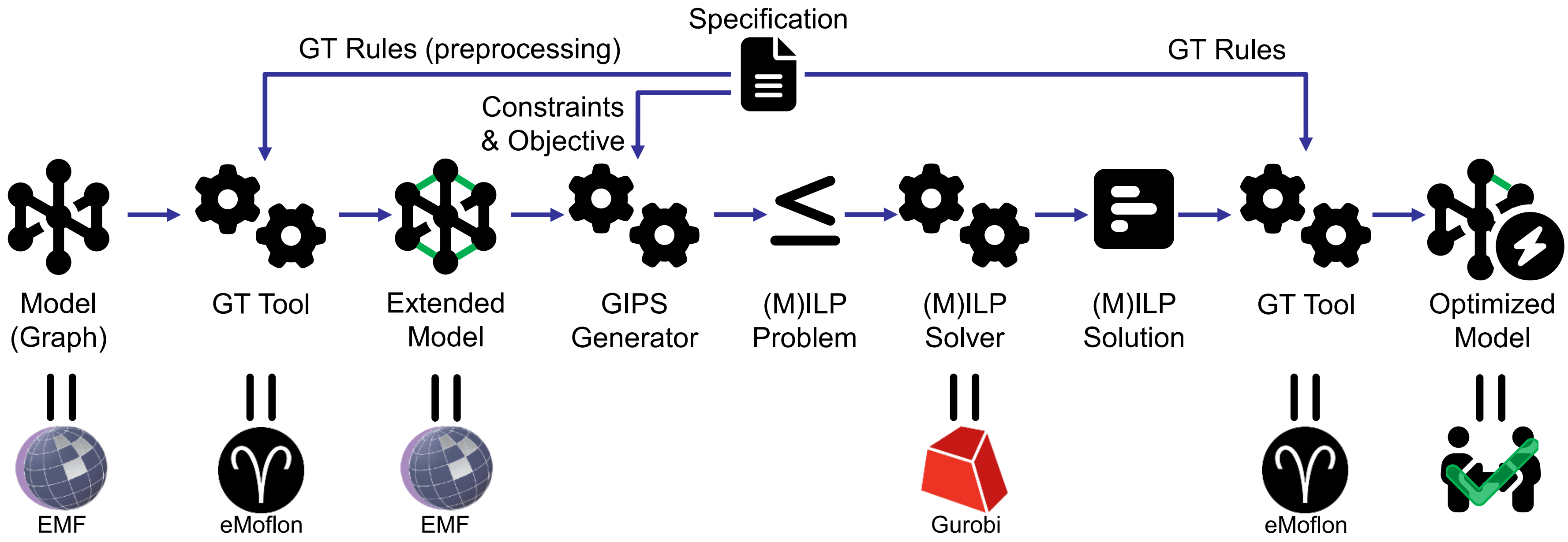
prev 0..1 0..1 next

Relaxation

(All icons from https://fontawesome.com)

The Graph-based (M)ILP Problem Specification Approach → GIPS

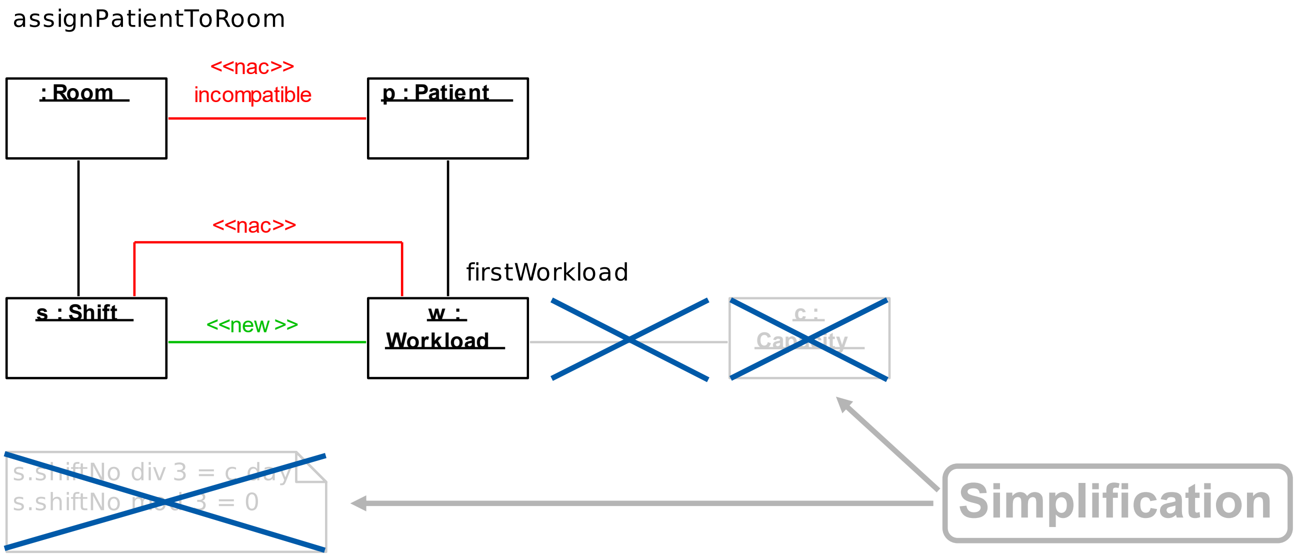
- GIPS Framework contains a domain-specific language **GIPSL**



(All icons from https://fontawesome.com, https://gurobi.com, https://eclipse.dev/emf, and https://emoflon.org)

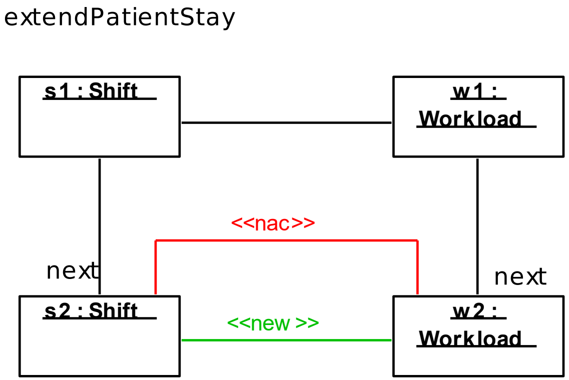
Preprocessing a Problem Instance (inital)

- Example: Assign a patient to a potential room



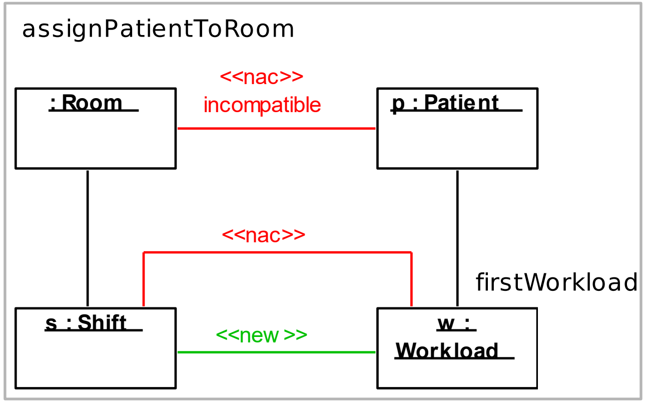
Preprocessing a Problem Instance (Extension)

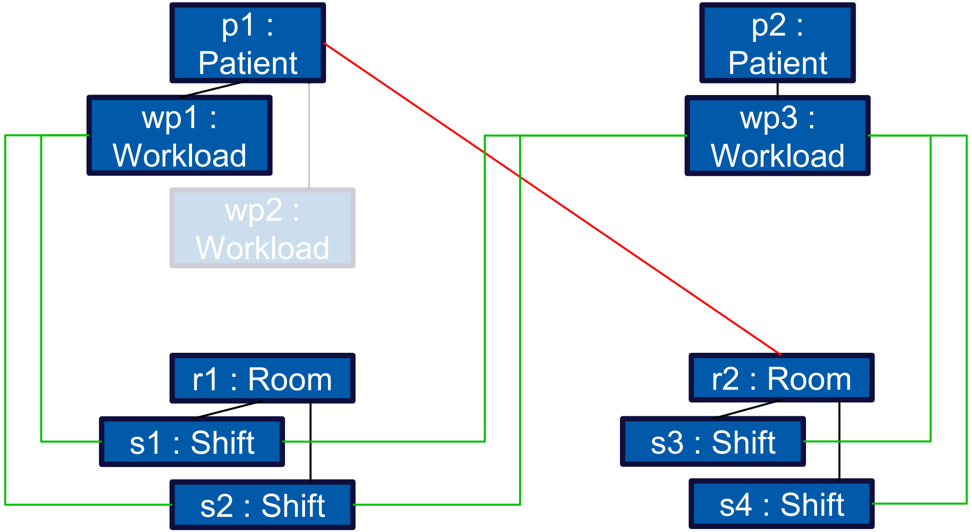
- Example: Extend a patient’s stay in a potential room



Preprocessing a Problem Instance (Application)

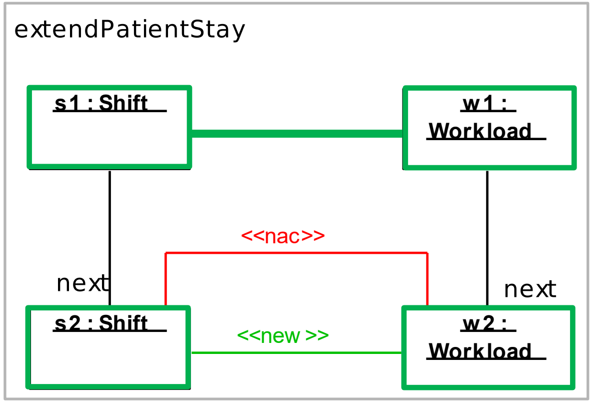
- Simplified example: Assign a patient to a potential room

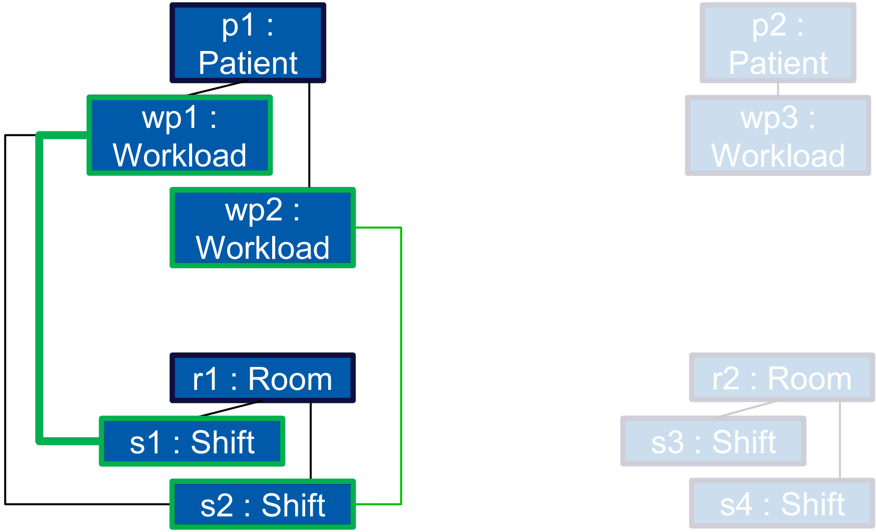




Preprocessing a Problem Instance (Application)

- Simplified example: Extend a patient’s stay in a potential room





Preprocessing a Problem Instance (Result)

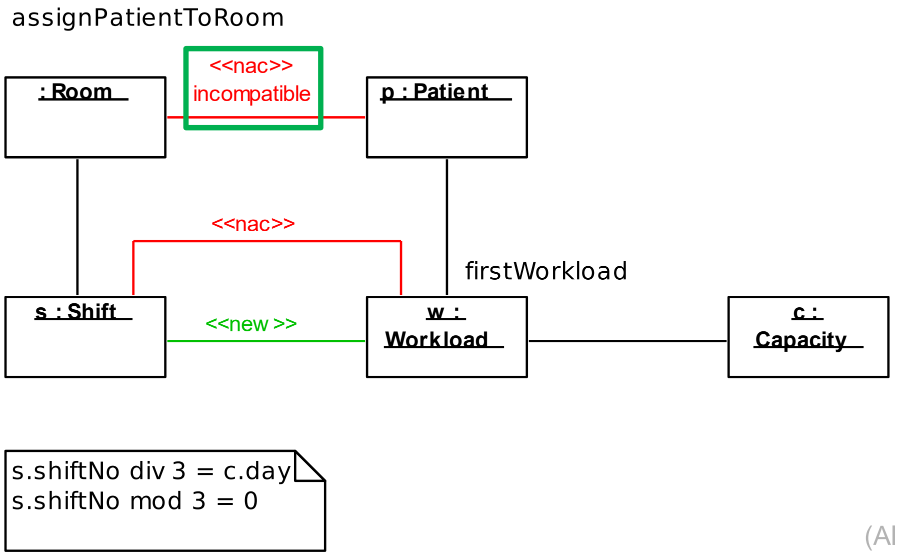
- The **Extended Model** contains all possible assignment edges
  - … for patients to rooms
  - … for surgeons to OTs & patients to OTs
  - … for nurses to rooms

? How to select the optimal assignment edges?

(All icons from https://fontawesome.com)

Constraint H2 (fulfilled by a GT Rule)

- *“Patients cannot be assigned to incompatible rooms.”*
  - Already enforced by the GT rule ✓

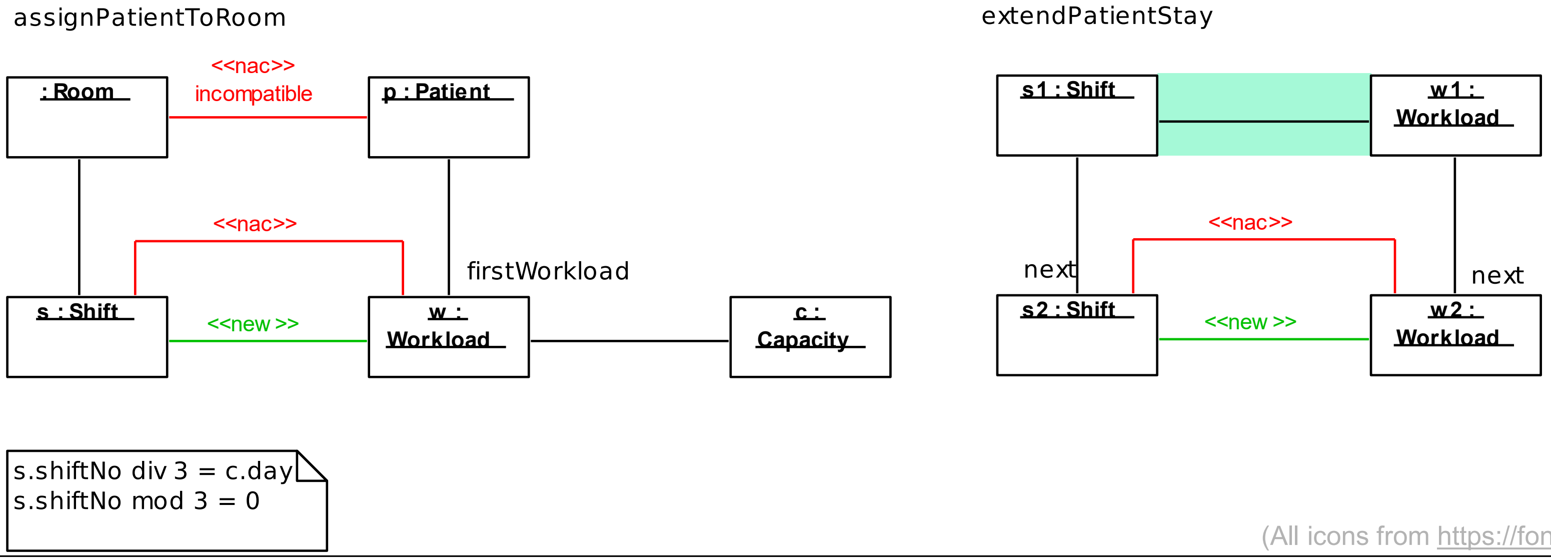


s.shiftNo div 3 = c.day
s.shiftNo mod 3 = 0

(All icons from https://fontawesome.com)

Constraint (fulfilled by a GT Rule)

- *“A patient is assigned to the same room during all shifts.”*
  - Already enforced by the GT rule(s) ✓

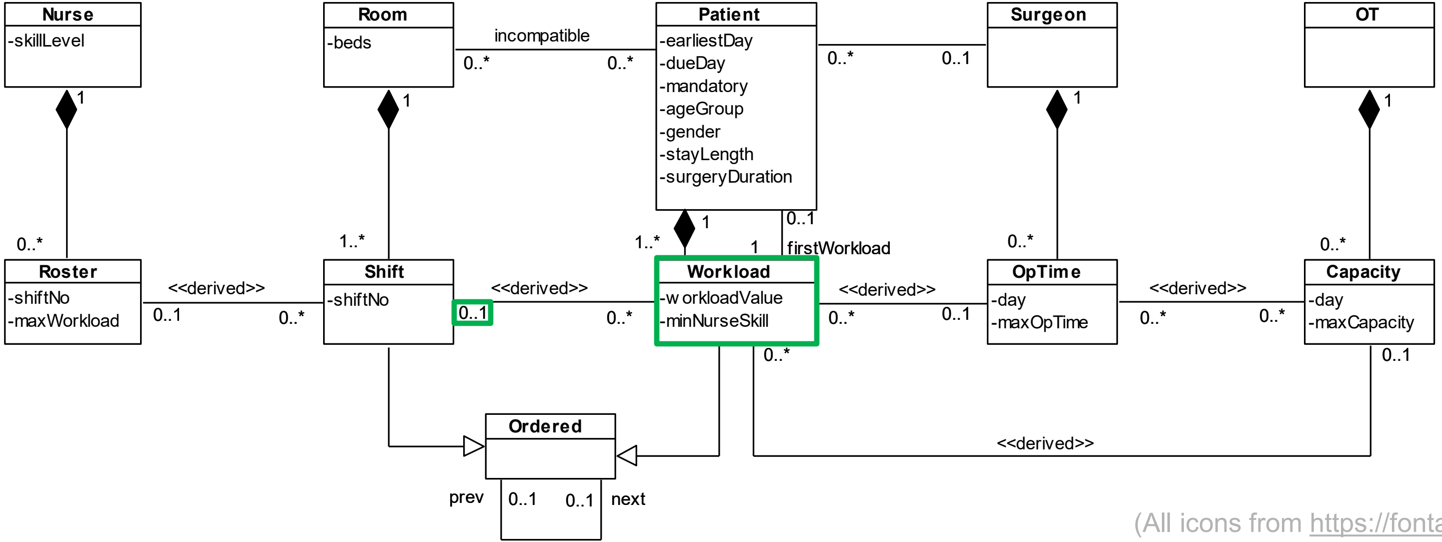


s.shiftNo div 3 = c.day
s.shiftNo mod 3 = 0

(All icons from https://fontawesome.com)

Constraint (fulfilled by the Metamodel)

- *“A patient is assigned to at most one room in each shift.”*
  - Already enforced by the (non-relaxed) metamodel. ✓

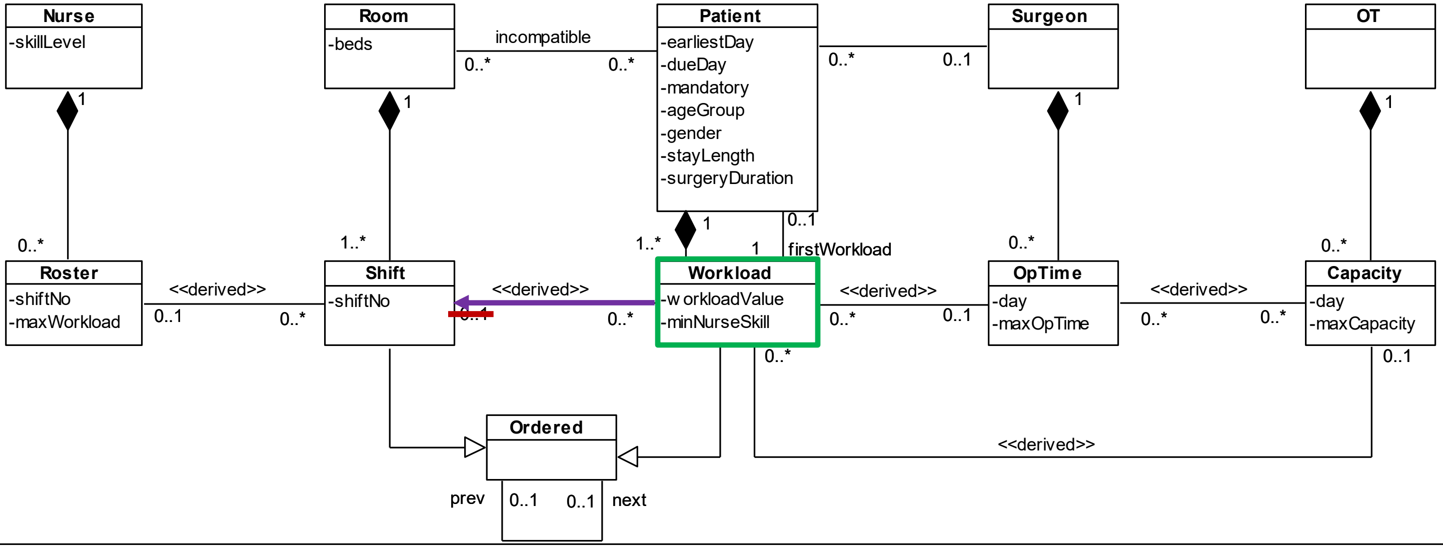


(All icons from https://fontawesome.com)

Constraint (relaxed Metamodel)

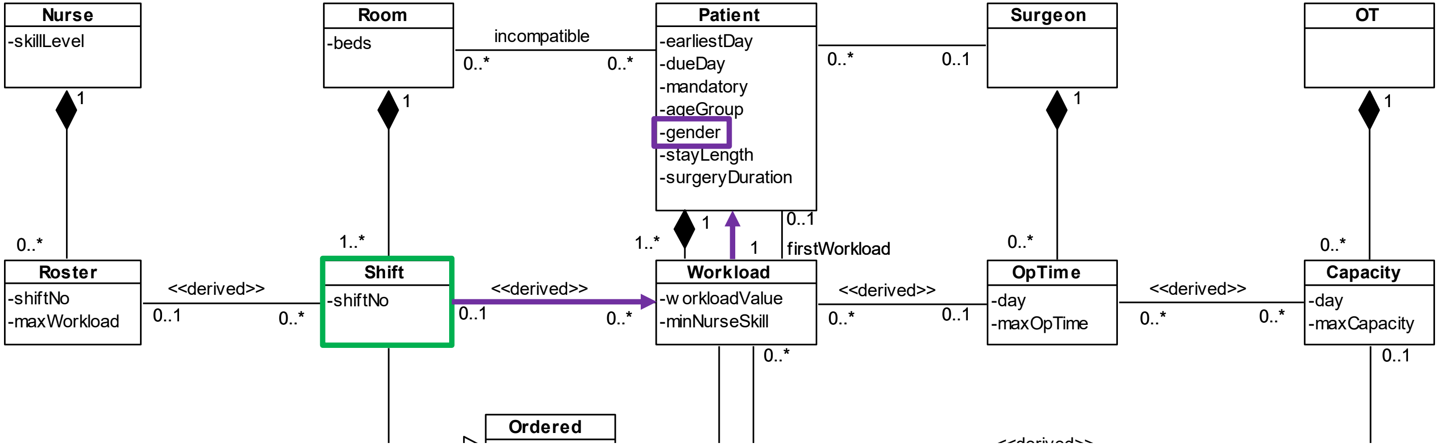
- *“A patient is assigned to at most one room in each shift.”*
  - `context Workload inv: shift->size() <= 1`

Object Constraint Language



Constraint H1 (for the Solver)

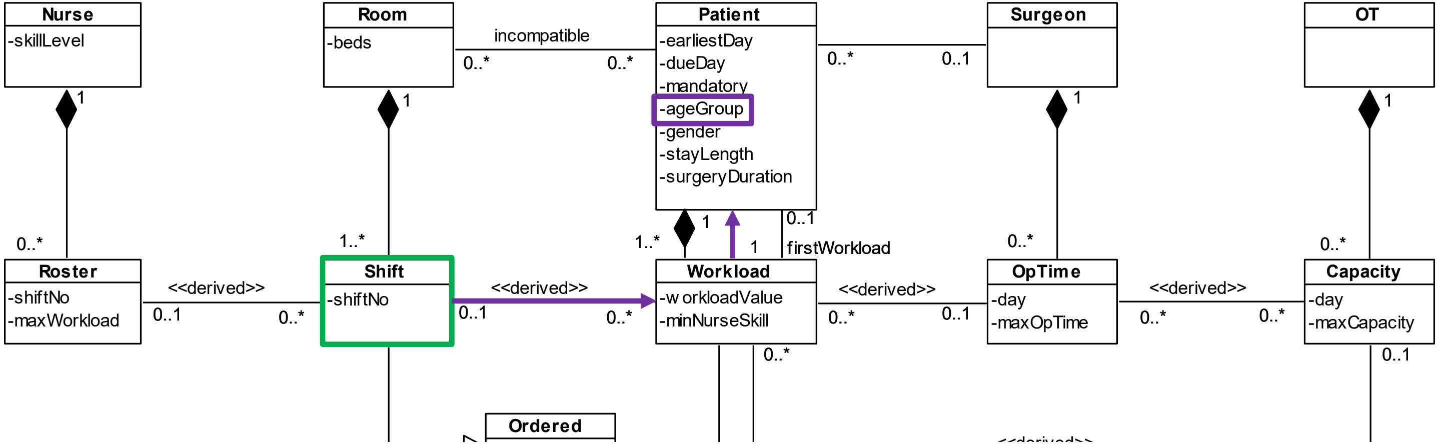
- *“No gender mix of patients in one room (in a given shift).”*
  - `context Shift inv:`
    `workload.patient.gender->asSet()->size() <= 1`



Constraint S1 (for the Solver)

- *“Maximum difference of age groups in one room should be minimized.”*

  -
    ```
    context Shift min: let ages = workload.patient.ageGroup in
      if ages->isEmpty() then 0 else (ages->max() - ages->min())
    ```



Submitted Solution – Our Steps

- Informal problem description (PDF)
- Metamodel + JSON import/export
- We derived the OCL constraints.
- We manually translated the OCL constraints to the GIPSL specification.
  - … hopefully automated in the future!

```
// [...]
// Mapping that is true if gender `g` is contained in room `r` on day `d`
mapping roomDayGender to dayRoomGender;

// H1: No gender mix per room
constraint with dayRoomTuple {
    mappings.roomDayGender->filter(
        element.nodes.r == context.nodes.r & element.nodes.d ==
        context.nodes.d)->sum(element.value)
    <= 1
}

constraint with roomDayGender {
    [mappings.roomDayPatientLoad->filter(
        element.nodes.d == context.nodes.d
        & element.nodes.r == context.nodes.r
        & element.nodes.p.gender == context.nodes.g.name
    )->sum(element.value)
    + patterns.occupantRoomDay->filter(
        element.nodes.d == context.nodes.d
        & element.nodes.r == context.nodes.r
        & element.nodes.o.gender == context.nodes.g.name
    )->sum(1)
    >= 1]
    => [context.value == 1]
}
// [...]
```

Resulting (example) LP file

```
// [...]
// Mapping that is true if gender `g` is contained in room `r` on day `d`
mapping roomDayGender to dayRoomGender;

// H1: No gender mix per room
constraint with dayRoomTuple {
    mappings.roomDayGender->filter(
        element.nodes.r == context.nodes.r & element.nodes.d ==
        context.nodes.d)->sum(element.value)
    <= 1
}

constraint with roomDayGender {
    [mappings.roomDayPatientLoad->filter(
        element.nodes.d == context.nodes.d
        & element.nodes.r == context.nodes.r
        & element.nodes.p.gender == context.nodes.g.name
    )->sum(element.value)
    + patterns.occupantRoomDay->filter(
        element.nodes.d == context.nodes.d
        & element.nodes.r == context.nodes.r
        & element.nodes.o.gender == context.nodes.g.name
    )->sum(1)
    >= 1]
    => [context.value == 1]
}
// [...]
```

→

```
Subject To
DisjunctMappingConstraint0OnroomDayGender_0:
- roomDayGender#6->DisjunctMappingConstraint0OnroomDayGender_symbolic1#1
+ roomDayGender#6->DisjunctMappingConstraint0OnroomDayGender_symbolic0#17
>= 0
DisjunctMappingConstraint0OnroomDayGender_1:
- roomDayGender#5->DisjunctMappingConstraint0OnroomDayGender_symbolic1#2
+ roomDayGender#5->DisjunctMappingConstraint0OnroomDayGender_symbolic0#18
>= 0
DisjunctMappingConstraint0OnroomDayGender_2:
- roomDayGender#2->DisjunctMappingConstraint0OnroomDayGender_symbolic1#4
+ roomDayGender#2->DisjunctMappingConstraint0OnroomDayGender_symbolic0#20
>= 0
DisjunctMappingConstraint0OnroomDayGender_3:
- roomDayGender#1->DisjunctMappingConstraint0OnroomDayGender_symbolic1#5
+ roomDayGender#1->DisjunctMappingConstraint0OnroomDayGender_symbolic0#21
>= 0
DisjunctMappingConstraint0OnroomDayGender_4:
- roomDayGender#0->DisjunctMappingConstraint0OnroomDayGender_symbolic1#0
+ roomDayGender#0->DisjunctMappingConstraint0OnroomDayGender_symbolic0#16
>= 0
DisjunctMappingConstraint0OnroomDayGender_5:
- roomDayGender#7->DisjunctMappingConstraint0OnroomDayGender_symbolic1#3
+ roomDayGender#7->DisjunctMappingConstraint0OnroomDayGender_symbolic0#19
>= 0
DisjunctMappingConstraint0OnroomDayGender_6:
- roomDayGender#3->DisjunctMappingConstraint0OnroomDayGender_symbolic1#7
+ roomDayGender#3->DisjunctMappingConstraint0OnroomDayGender_symbolic0#23
>= 0
```

Conclusion and Future Work

- ~2 weeks of rapid prototyping.
- We solved all instances in 10 min.
- We were unable to specify all (soft) constraints.
  - Because of the (implementation) time limit ☹
  - Because of the runtime requirement (10 min. time limit) ☹

- Finish the prototype implementation.
- Evaluate all hidden instances.
- Performance tuning.
- Write a paper about our solution ☺

Thank you for your attention.

Any questions?

My contact information:

(All icons from https://fontawesome.com)

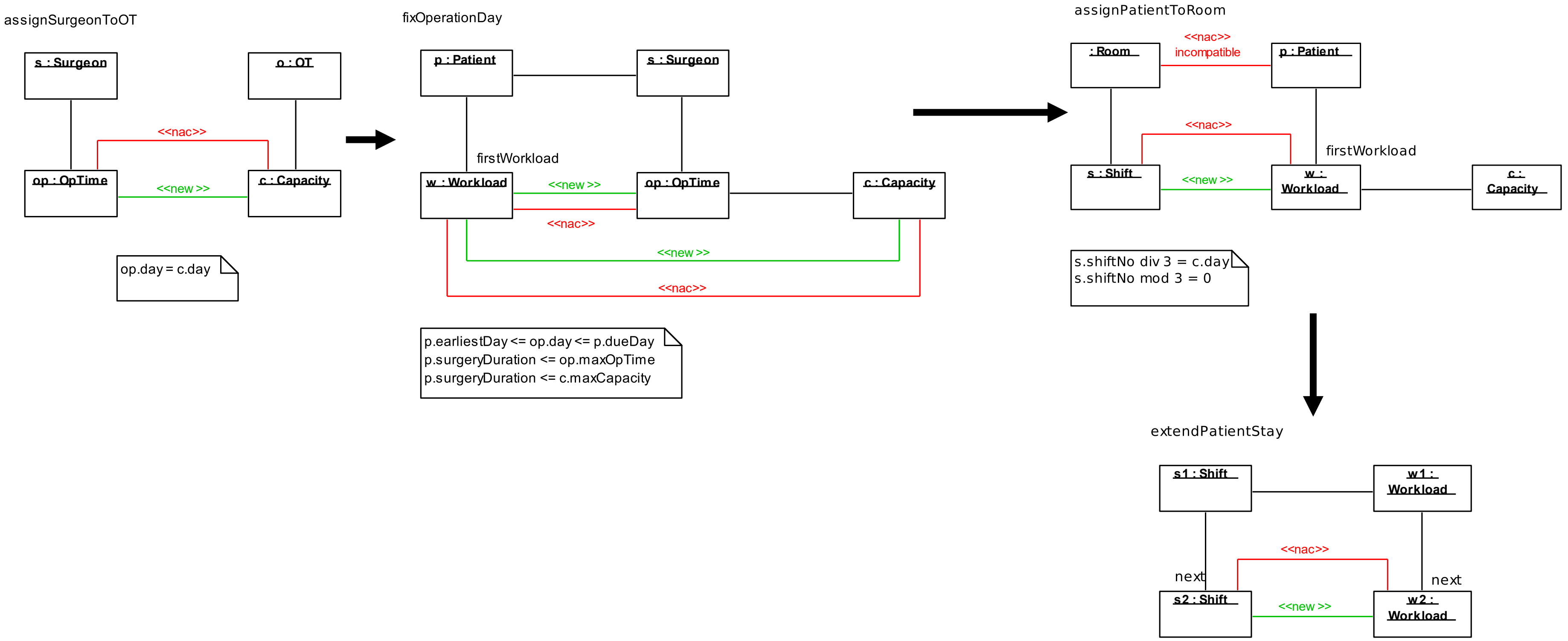
Backup Slide: OCL Auxilary Operations

- `set->max() := set->iterate(element:Integer; result: Integer=0 | if element > result then element else result)`

- `set->min() := set->iterate(element:Integer; result: Integer=maxInt | if element < result then element else result)`

- `diff(x,y) := if x > y then x-y else 0`

Backup Slide: GT Rule Dependencies



op.day = c.day

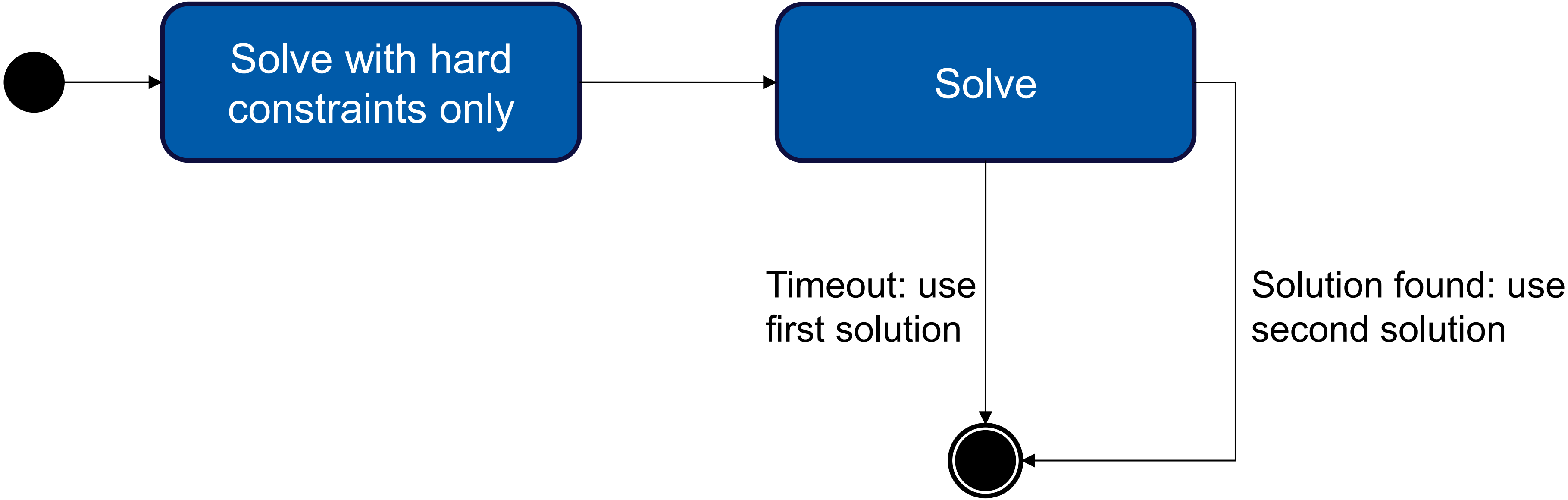
p.earliestDay <= op.day <= p.dueDay
p.surgeryDuration <= op.maxOpTime
p.surgeryDuration <= c.maxCapacity

s.shiftNo div 3 = c.day
s.shiftNo mod 3 = 0

Backup Slide: GIPSL Specification vs. LP File

File	Lines of Code (LoC)
GIPSL spec. (hard constraints only)	337
GIPSL spec. (complete)	374
LP file (to solve `*test01.json*`)	34.833

Backup Slide: Pipeline Approach


Solve with hard constraints only
Solve
Timeout: use first solution
Solution found: use second solution