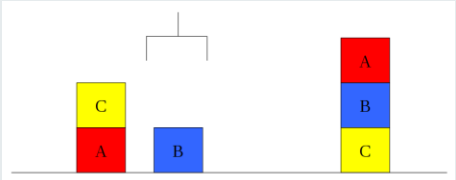
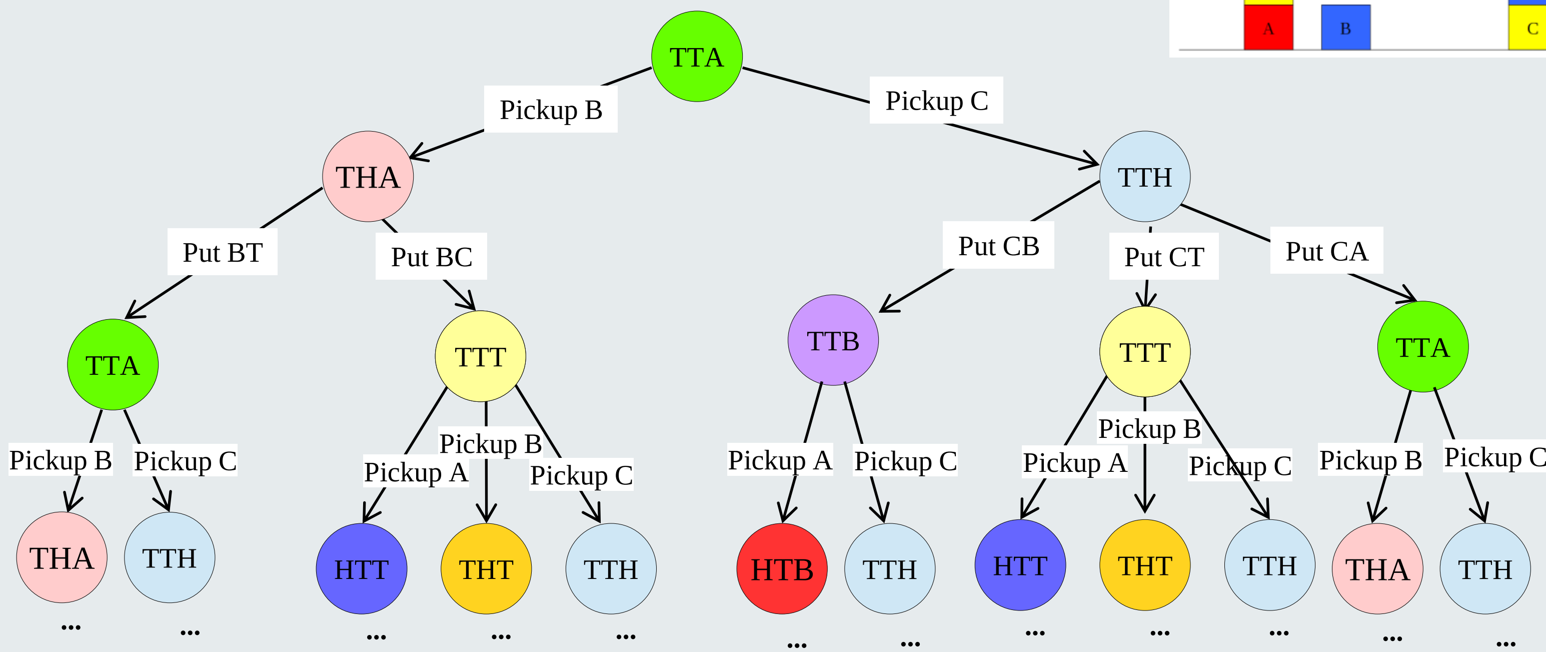
Expressive Planning by Combining Forward Search and Mixed-Integer Programming

Elad Denenberg

King's College London

Classical Planning as Forward Search





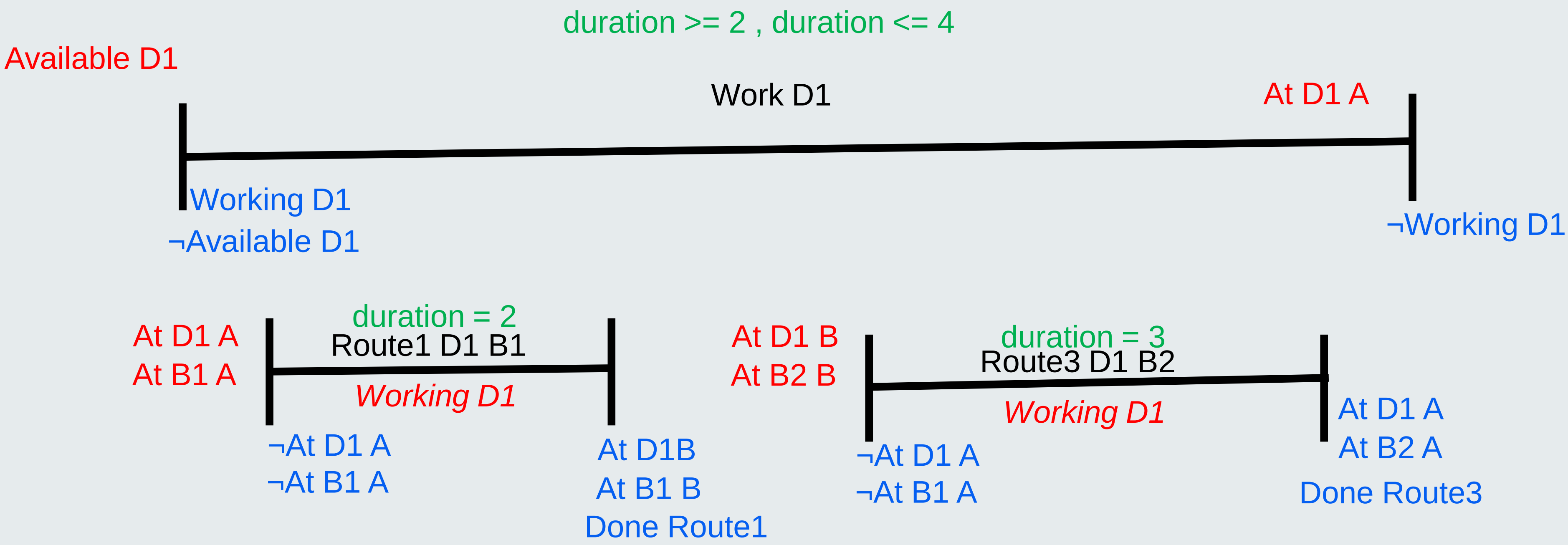
Classical Planning Gives Us:

- Propositional Relaxation Heuristics: RPG, Causal Graph;
- Search Guidance: Helpful Actions/Preferred Operators.
- Search Techniques, Enforced Hill-Climbing, Multi Open List Search, Memoisation;

Running Example – Public Transport

- Drivers have working hours;
- Bus routes have fixed durations and start and end locations.
- Goals are that each bus route is done.
- The routes have timetables that they must follow.

Temporal Planning: Public Transport



Conditions and Effects at the start and at the end;

Invariant/overall conditions;

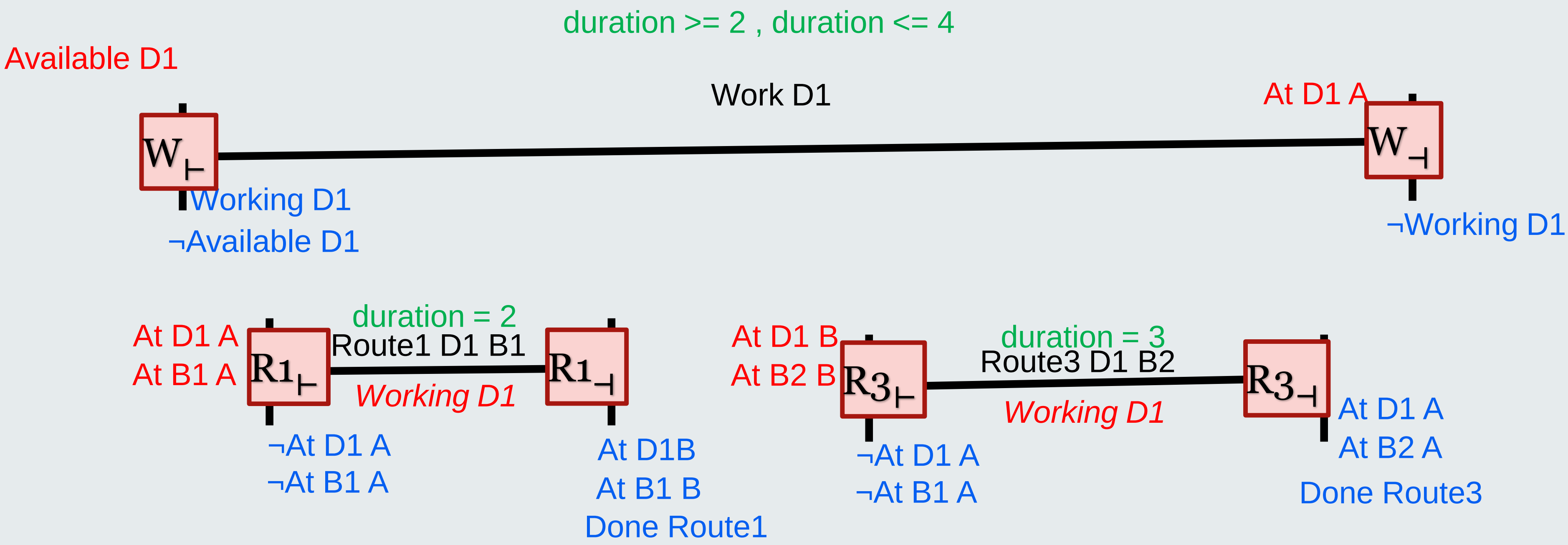
Durations constraints:

(= ?duration 4)

(and (>= ?duration 2) (<= ?duration 4))

"Planning with Problems Requiring Temporal Coordination." A. I. Coles, M. Fox, D. Long, and A. J. Smith. AAAI 2008.
"Managing concurrency in temporal planning using planner-scheduler interaction." A. I. Coles, M. Fox, K. Halsey, D. Long, and A. J. Smith. Artificial Intelligence. 173 (1) (2009).

Temporal Planning: Public Transport

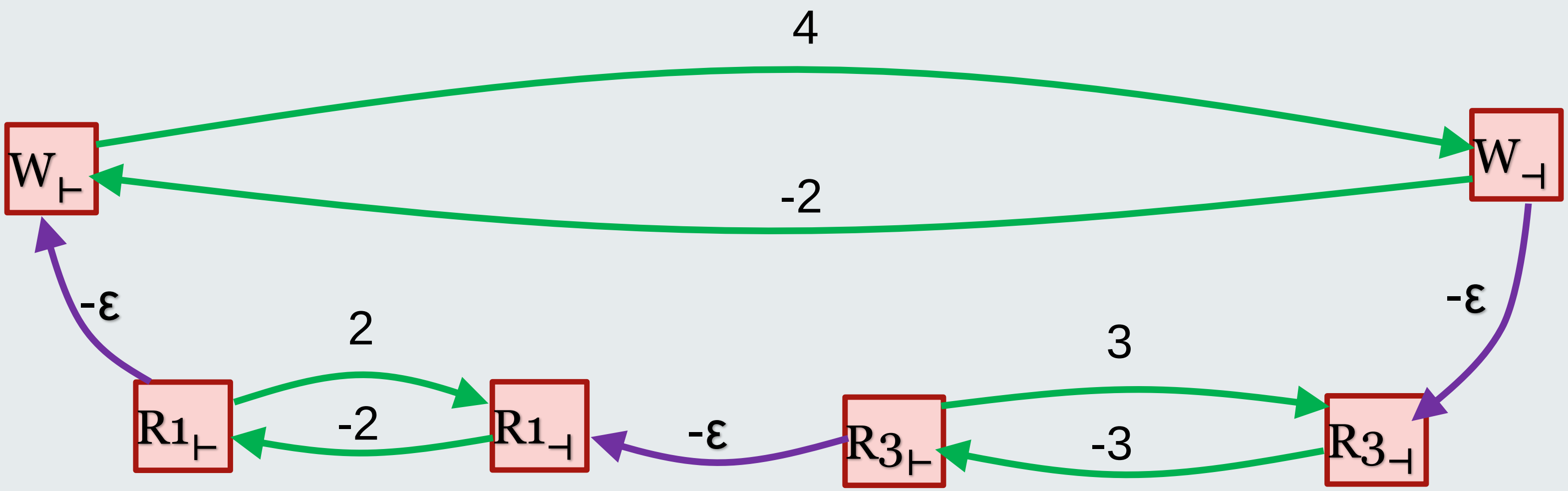


Three Challenges:

- Make sure ends can't be applied unless starts have.
- Overall Conditions.
- Duration constraints.

"Planning with Problems Requiring Temporal Coordination." A. I. Coles, M. Fox, D. Long, and A. J. Smith. AAAI 2008.
"Managing concurrency in temporal planning using planner-scheduler interaction." A. I. Coles, M. Fox, K. Halsey, D. Long, and A. J. Smith. Artificial Intelligence. 173 (1) (2009).

Temporal Planning: Public Transport



Constraints:

$W_{\dashv} - W_{\vdash} >= 2$

$W_{\dashv} - W_{\vdash} <= 4$

$R1_{\vdash} >= W_{\vdash} + \varepsilon$

$R1_{\dashv} - R1_{\vdash} = 2$

$R3_{\vdash} >= R1_{\vdash} + \varepsilon$

$R3_{\dashv} - R3_{\vdash} = 3$

$W_{\dashv} >= R3_{\dashv} + \varepsilon$

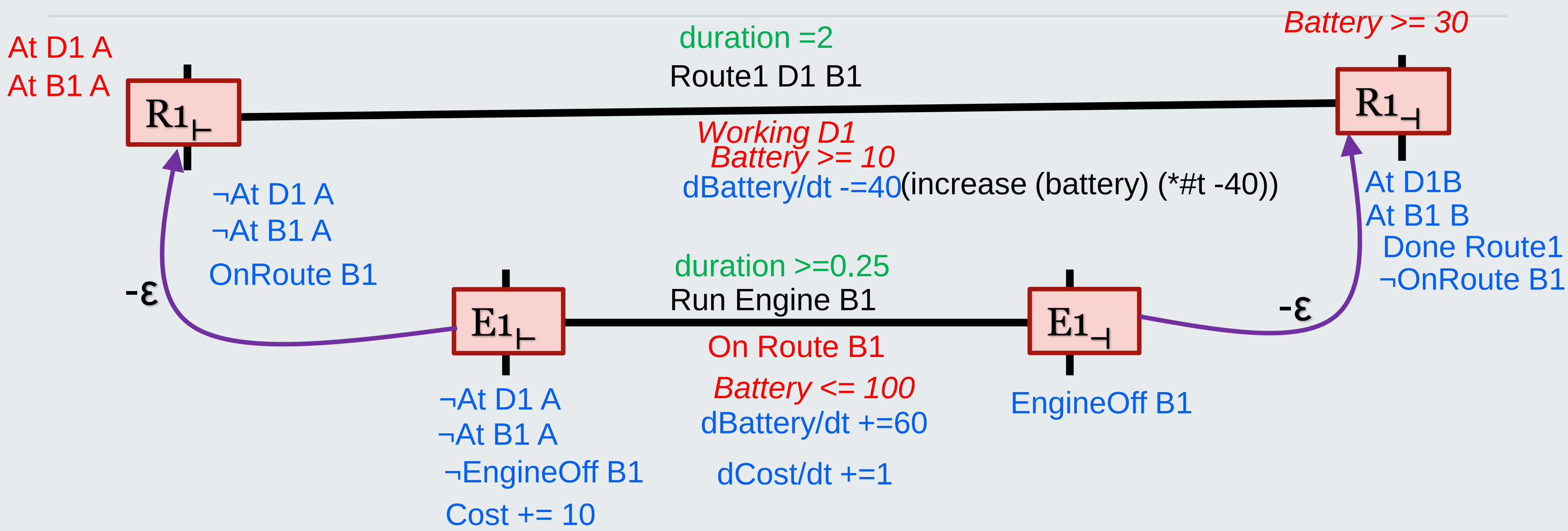
"Planning with Problems Requiring Temporal Coordination." A. I. Coles, M. Fox, D. Long, and A. J. Smith. AAAI 2008.
"Managing concurrency in temporal planning using planner-scheduler interaction." A. I. Coles, M. Fox, K. Halsey, D. Long, and A. J. Smith. Artificial Intelligence. 173 (1) 2009.

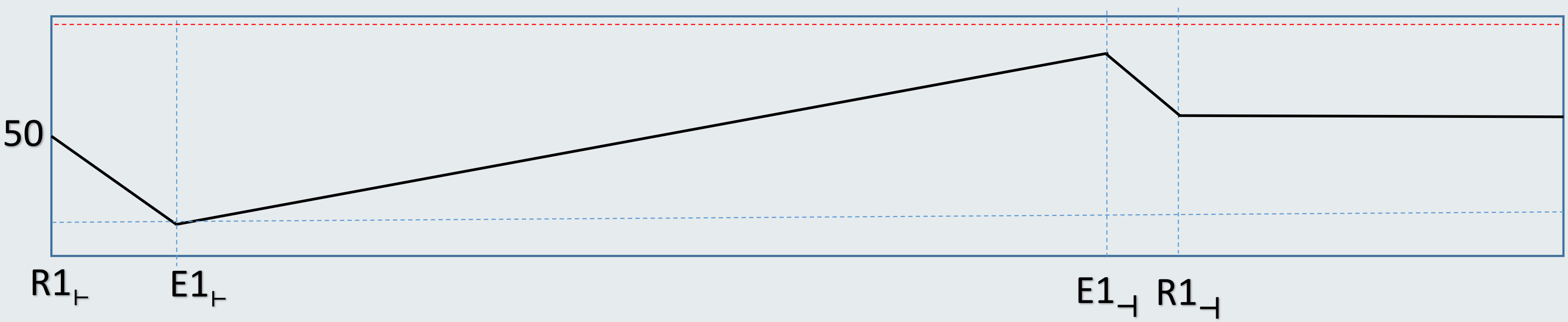
Continuous Linear Change

- Numeric quantities so far we have seen change instantaneously:
  - e.g. `(at end (decrease (battery) 1))`
  - **Or** `(at end (decrease (battery) (+ (*3 (?duration)) (*0.5 (temperature))))`
  - **V' = W . V + C**

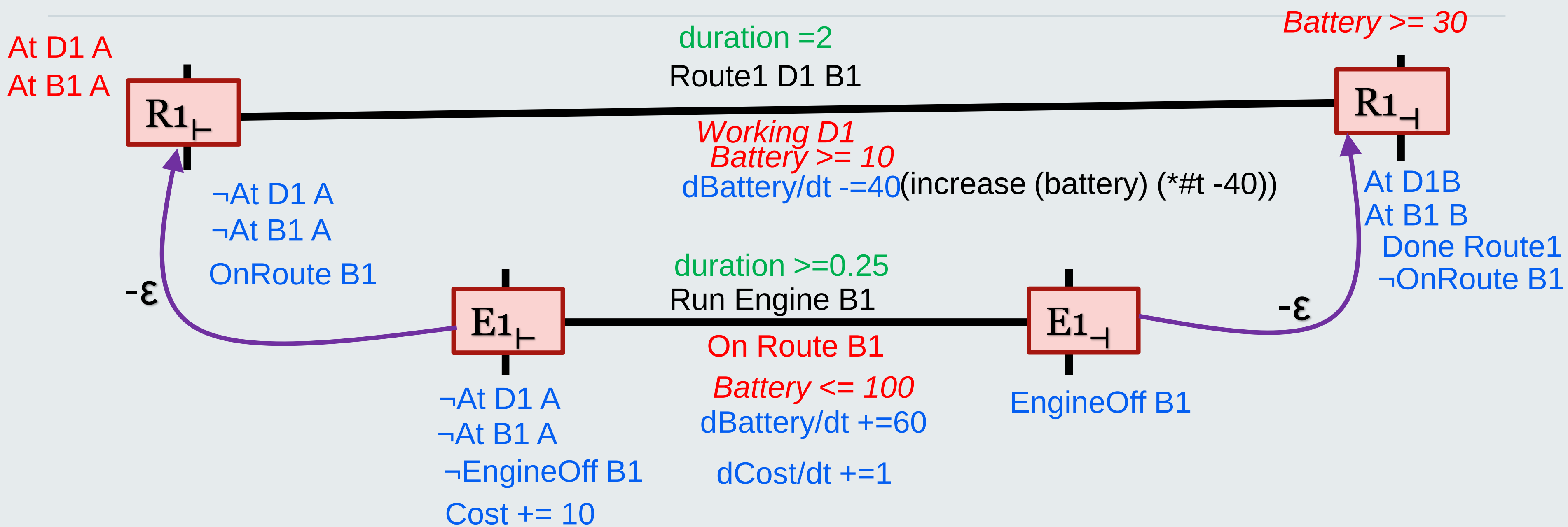
- In reality numeric values often change continuously, rather than discretely.
  - While the bus is running $\frac{d\,battery}{dt}$=-40
  - e.g. `(increase (battery) (*#t -40))`
  - Today we will deal with linear change only.

Continuous Linear Change: Colin





Continuous Linear Change: Colin



Temporal Constraints:

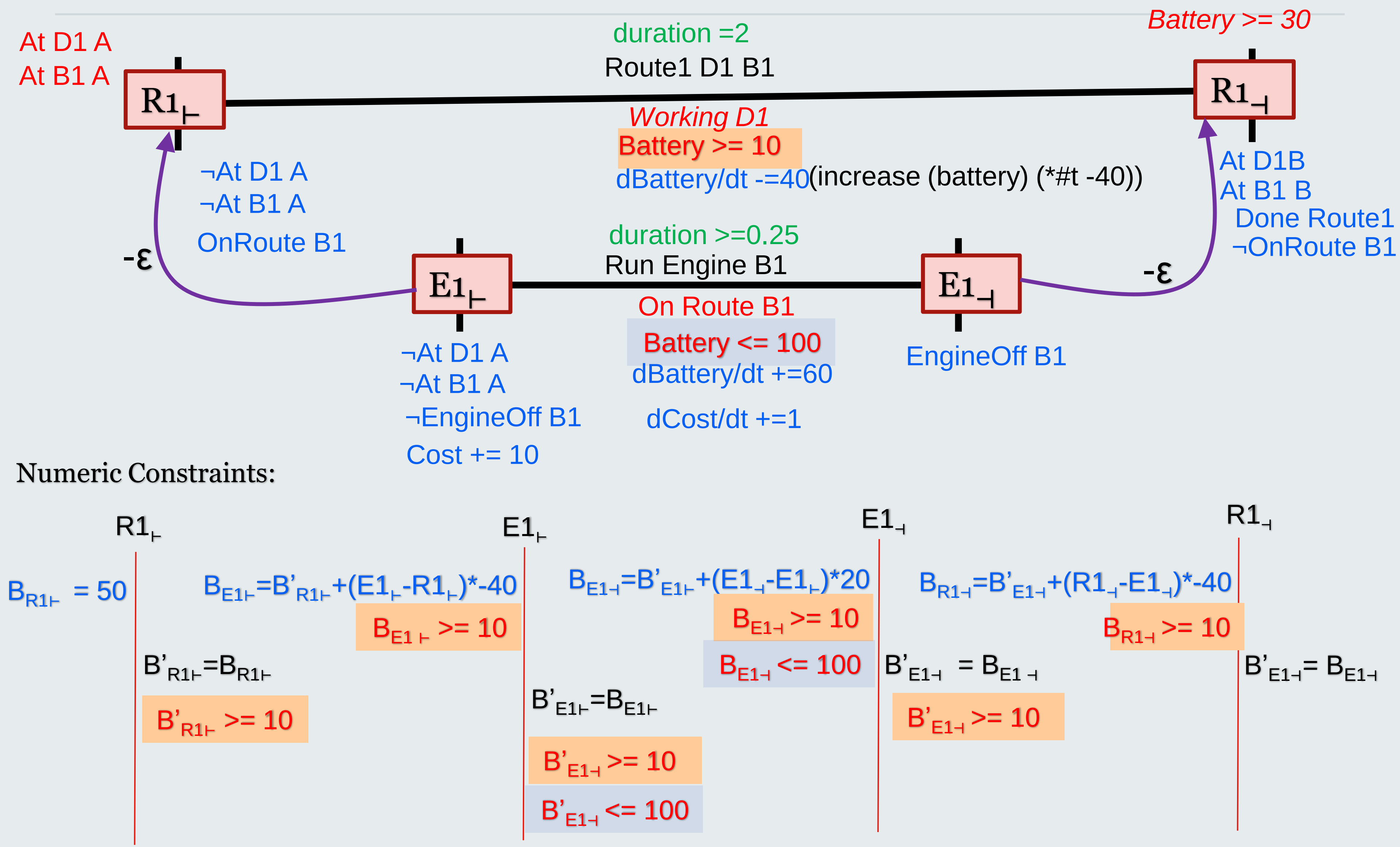
$R1_{\dashv} - R_{\vdash} = 2$

$E1_{\vdash} >= W_{\vdash} + \varepsilon$

$E1_{\dashv} - E1_{\vdash} >= 0.25$

$R1_{\vdash} >= E1_{\vdash} + \varepsilon$

Continuous Linear Change: Colin



"Temporal Planning in Domains with Linear Processes." A. J. Coles, A. I. Coles, M. Fox, and D. Long. IJCAI (2009).
"COLIN: Planning with Continuous Linear Numeric Change." A. J. Coles, A. I. Coles, M. Fox and D. Long. JAIR (44) (2013)

Writing The LP

- For each (snap) action, A_i, in the (partial) plan create the following LP variables for each numeric variable in the problem:
    - v_i: the value of that variable immediately before A_i is executed;
    - v'_i: the value v immediately after A_i is executed.
    - δv_i: the rate of change active on v after A_i is executed.

- Create a single LP variable t_i to represent the time at which A_i will be executed.

Writing the LP - Constraints

- Initial values:
  - v_o = initial state value of v;
- Temporal Constraints:
  - $t_i >= t_{i-1} + \varepsilon$
  - $t_j - t_i <= max\_dur\ A$ (where t_j is the end of the action starting at t_i)
  - $t_j - t_i >= min\_dur\ A$ (where t_j is the end of the action starting at t_i)
- Continuous Change
  - $v_{i+1} = v'_i + \delta v_i (t_{i+1} - t_i)$
- Discrete Change:
  - $v'_i = v_i + \mathbf{w} . \mathbf{v_i}$;
  - e.g. : $v'_i = v_i + 2\ u_i - 3w_i$

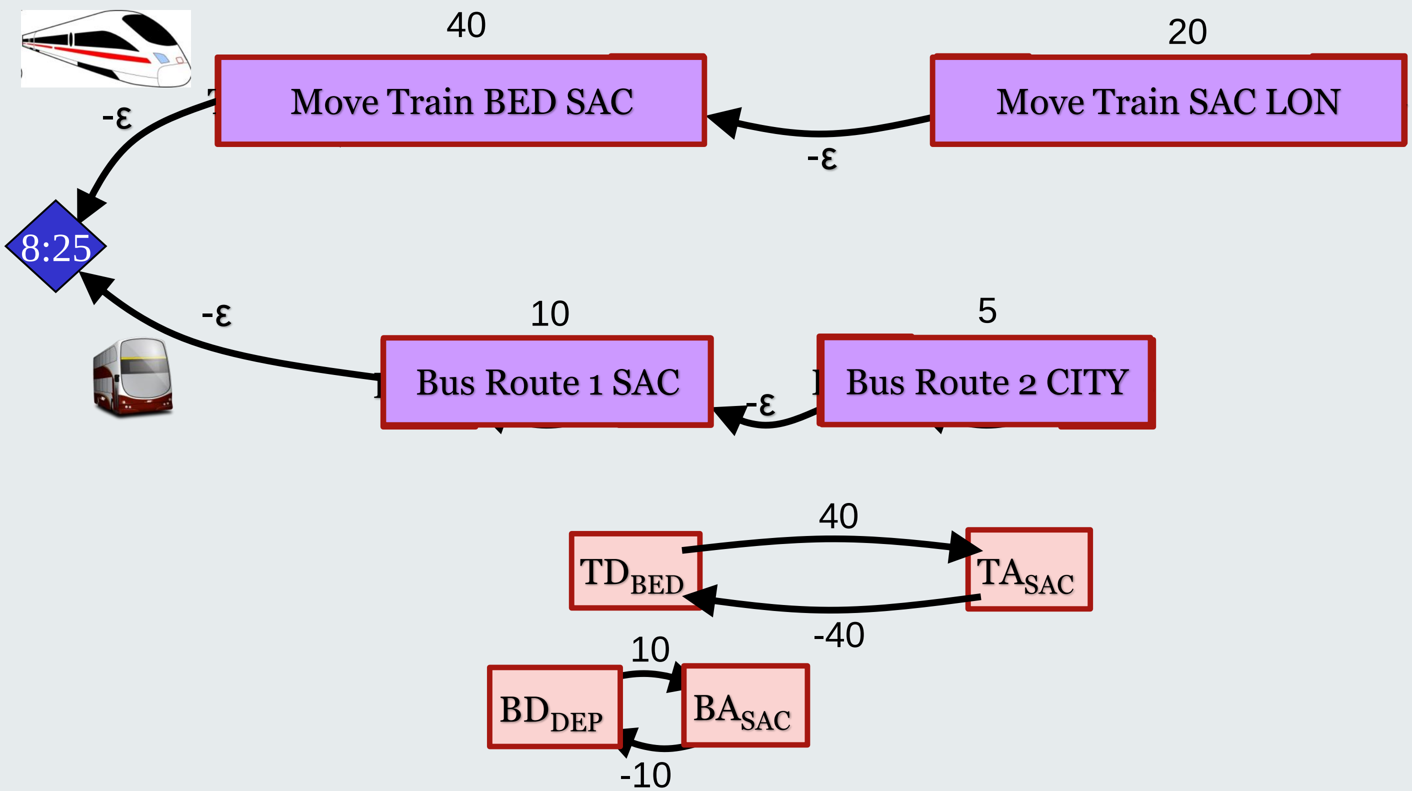
Writing the LP - Constraints

- Preconditions: constraints over v_i:
  - $\mathbf{w} . \mathbf{v_i}$ {>=,=,<=} c;
  - e.g. 2wi -3ui <= 4;
- Invariants of A, must be checked before and after every step between the start (i) and end (j) of A.
  - $\mathbf{w} . \mathbf{v'_i}$ {>=,=,<=} c;
  - $\mathbf{w} . \mathbf{v_{i+1}}$ {>=,=,<=} c;
  - $\mathbf{w} . \mathbf{v'_{i+1}}$ {>=,=,<=} c;
  - …
  - $\mathbf{w} . \mathbf{v_j}$ {>=,=,<=} c;

Writing the LP - Notes

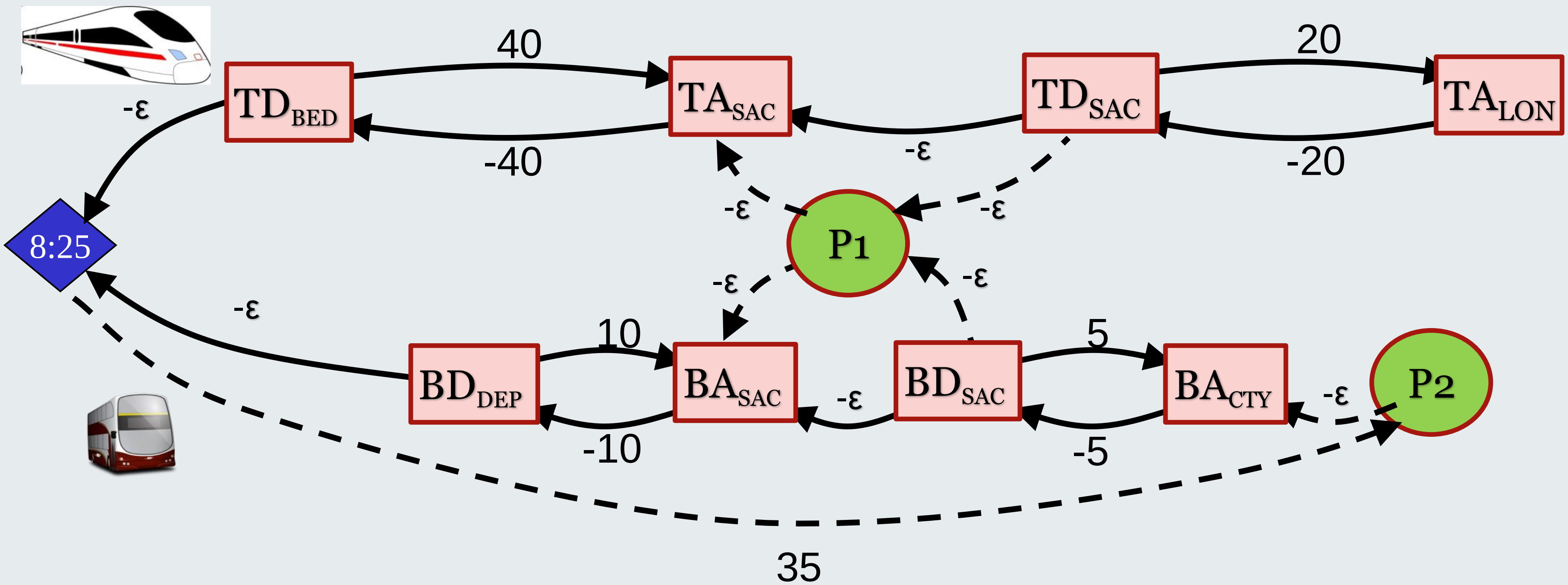
- This only works for linear change
- Defining an objective function will ensure cost optimality for the given state only
- Action applicability:
  - To check if the next action is applicable we need the bounds on the current variables
  - Define t_{now} and v_{now} , for the next action, use LP to maximise and minimise v_{now}

Partial Order Planning Forwards: POPF



"Forward-Chaining Partial-Order Planning." A. J. Coles, A. I. Coles, M. Fox, and D. Long. ICAPS 2010
"Have I Been Here Before? State Memoisation in Temporal Planning" A. J. Coles and A. I. Coles. ICAPS 2016.

Optimising Preferences: OPTIC and LPRPG-P



- The train and the bus are at the station simultaneously: (sometime (and (at train SAC) (at bus SAC)))

- The bus arrives in the city at 9am or earlier: (within 35 (at bus CITY)))

" Searching for Good Solutions in Goal-Dense Search Spaces." A. J. Coles and A. I. Coles. ICAPS (2013)
"Temporal Planning with Preferences and Time-Dependent Continuous Costs." J. Benton, A. J. Coles and A. I. Coles. ICAPS (2012)
"LPRPG-P: Relaxed Plan Heuristics for Planning with Preferences." A. J. Coles and A. I. Coles. ICAPS (2011)

Preferences

- Train arrives before bus departs:
  - BD_{SAC} - TA_{SAC} >= 0.01
- Bus arrives before train departs:
  - TD_{SAC} – BA_{SAC} >= 0.01
- Bus arrives at CTY by 9am (time 35):
  - BA_{CTY} <= 35

- But these are not hard constraints

- Use Big M constraints

Big M Constraints

- We need:
  - A 0/1 integer variable per preference p_1 , p_2
  - A very large constant M.
- Train arrives before bus departs:
  - $BD_{SAC} - TA_{SAC} + Mp_1 >= 0.01$
- Bus arrives before train departs:
  - $TD_{SAC} – BA_{SAC} + Mp_1 >= 0.01$
- Bus arrives at CTY by 9am (time 35):
  - $BA_{CTY} - Mp_2 <= 35$
- In the objective function:
  - Minimise: whatever + 5 p_1 + 2 p_2

Relationship Between Planners

- CRIKEY = FF + STN;
- Colin = CRIKEY s/STN/LP/;
- POPF = COLIN + Fewer ordering constraints;
- OPTIC = POPF + Preferences.

Planners Performance

- Heuristic computation is notoriously expensive:
  - An analysis showed that FF spends ~80% of its time evaluating the heuristic.
- COLIN:
  - Empirically using an STP scheduler scheduling accounts for on average less than 5% of state evaluation time.
  - For CLP and CPLEX (LP solvers) the figures are 13% and 18% respectively.
  - So better than calculating the heuristic.

Planners Performance Cont.

- OPTIC

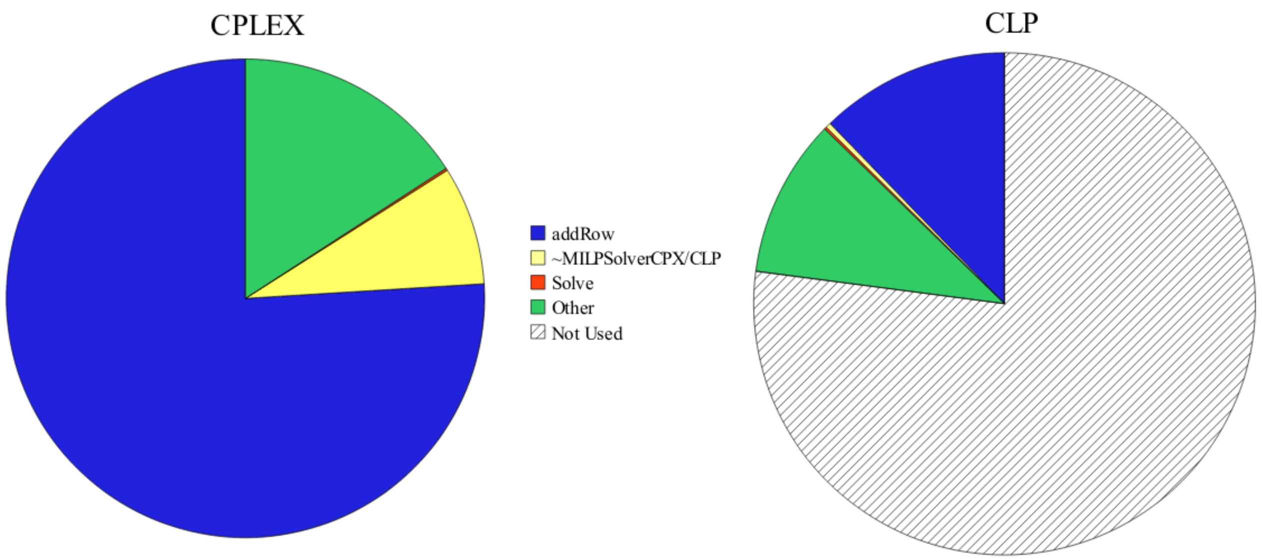


Figure 20: Time spent in various activities by each of the solvers, CPLEX and CLP, viewed as a proportion of the total time spent by CPLEX. The slice labelled '~MILPSolverCPX/CLP' is time spent in the destructor for the MILP solver in CPLEX or CLP: this is a housekeeping operation in the implementations (which are both written in C++).

Questions ?