Computational Optimization

Branch and Price and Check

Pascal Van Hentenryck

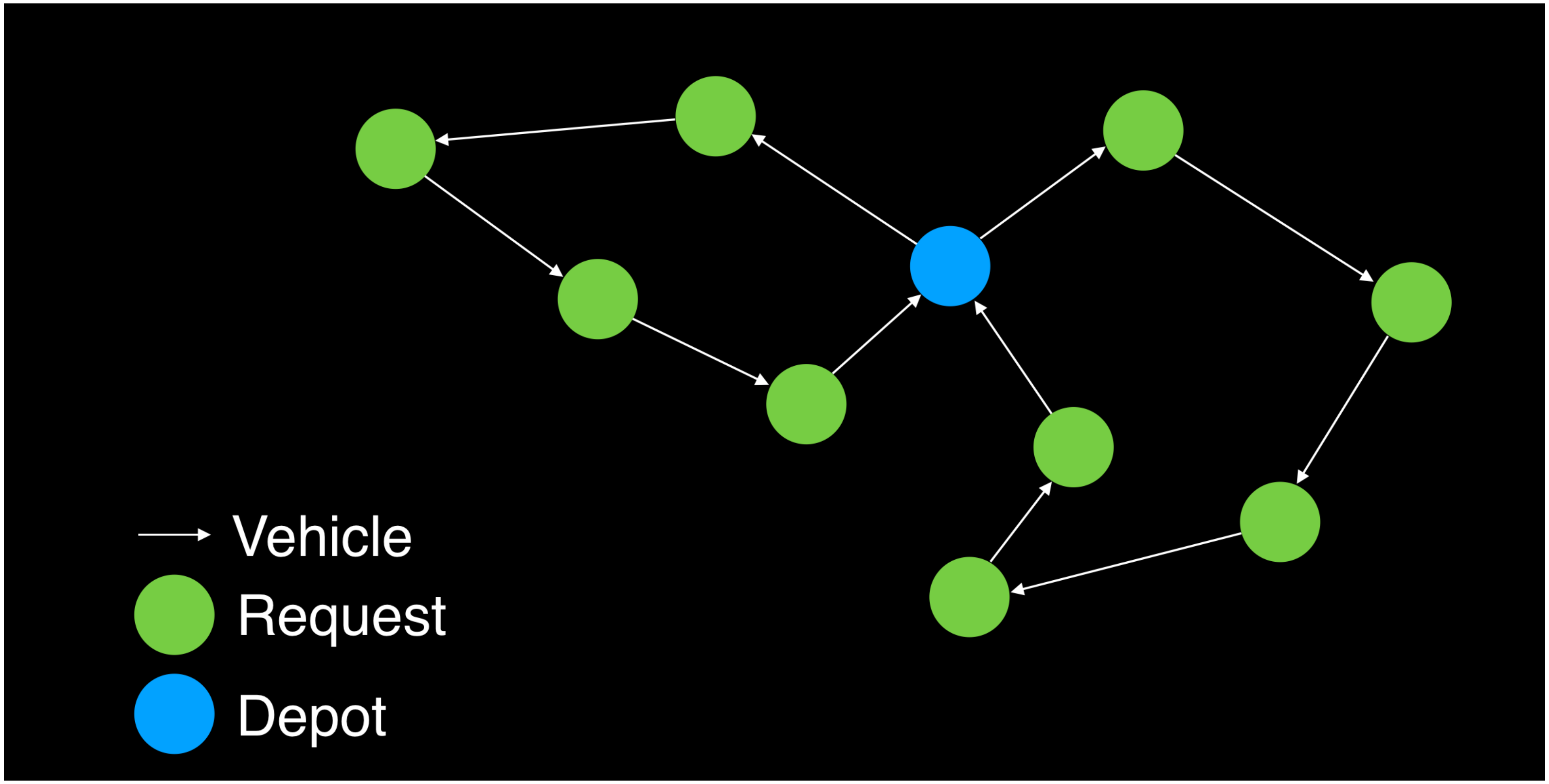
VRP with Location Constraints

- The problem
- Motivating applications
- Models
- Branch and Check and Price
- Experimental results

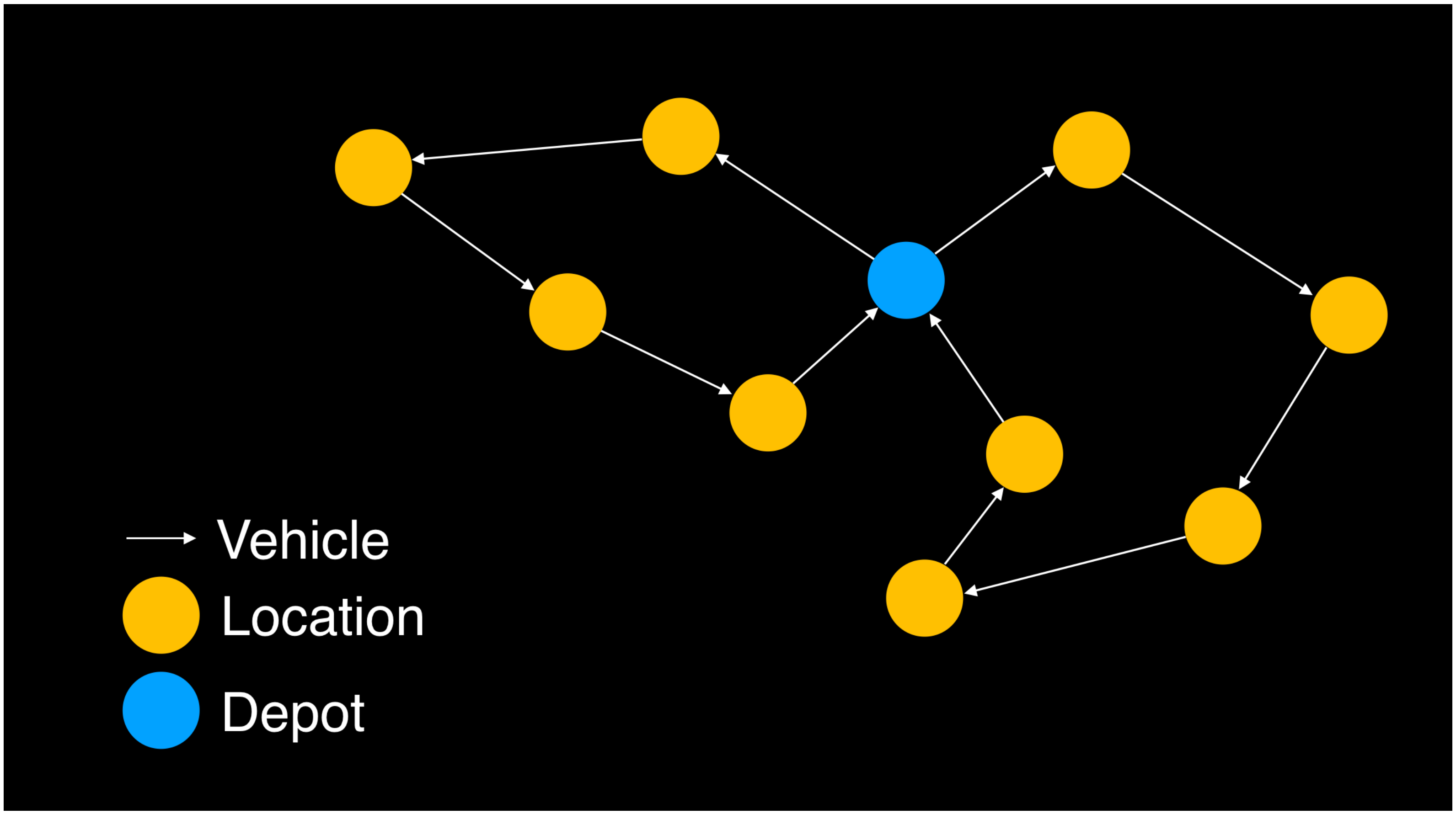
VRP with Location Constraints

- The problem
  - multiple vehicles
  - pickup and delivery constraints
  - time windows
  - capacity constraints
- Location constraints
  - number of parking slots at an airport
  - number of landings and takeoffs at an airport in a given intervals

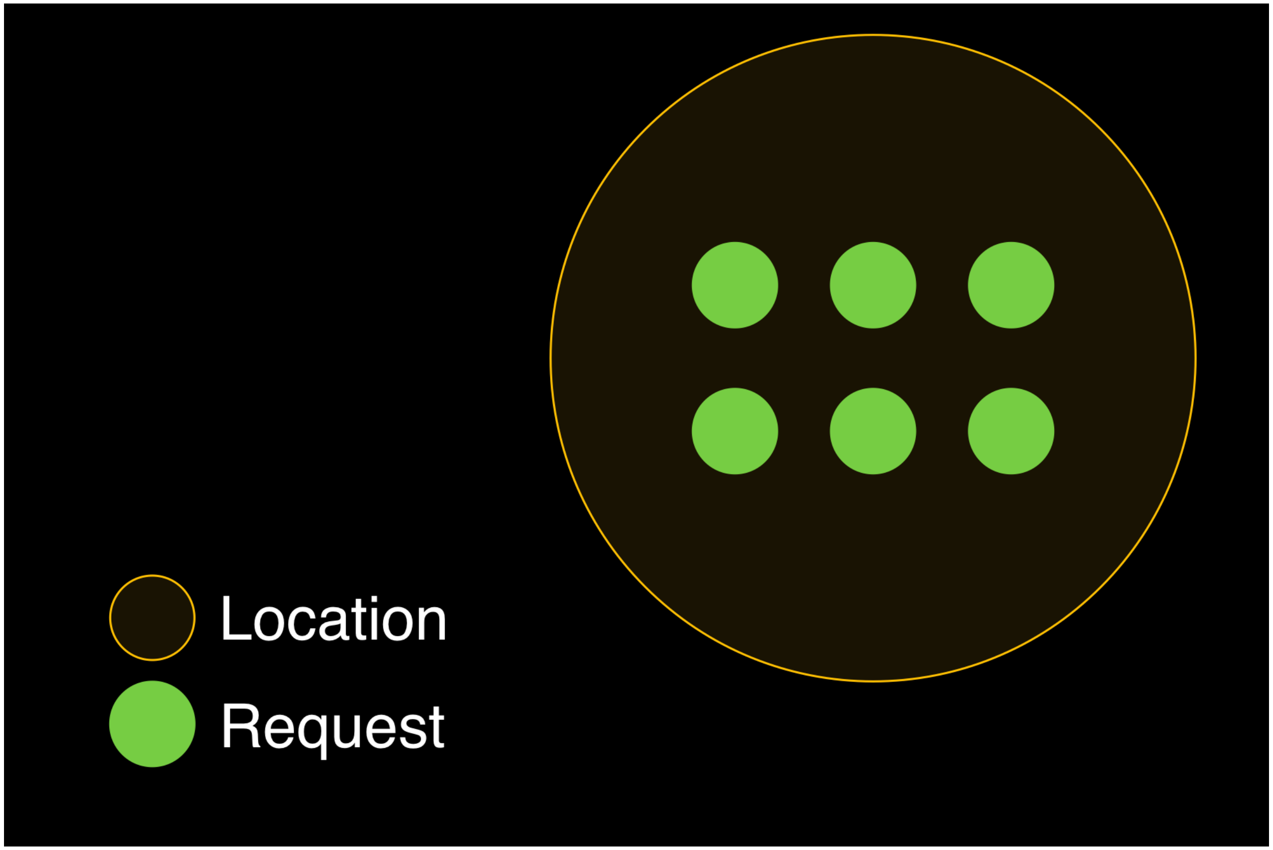
Traditional VRP



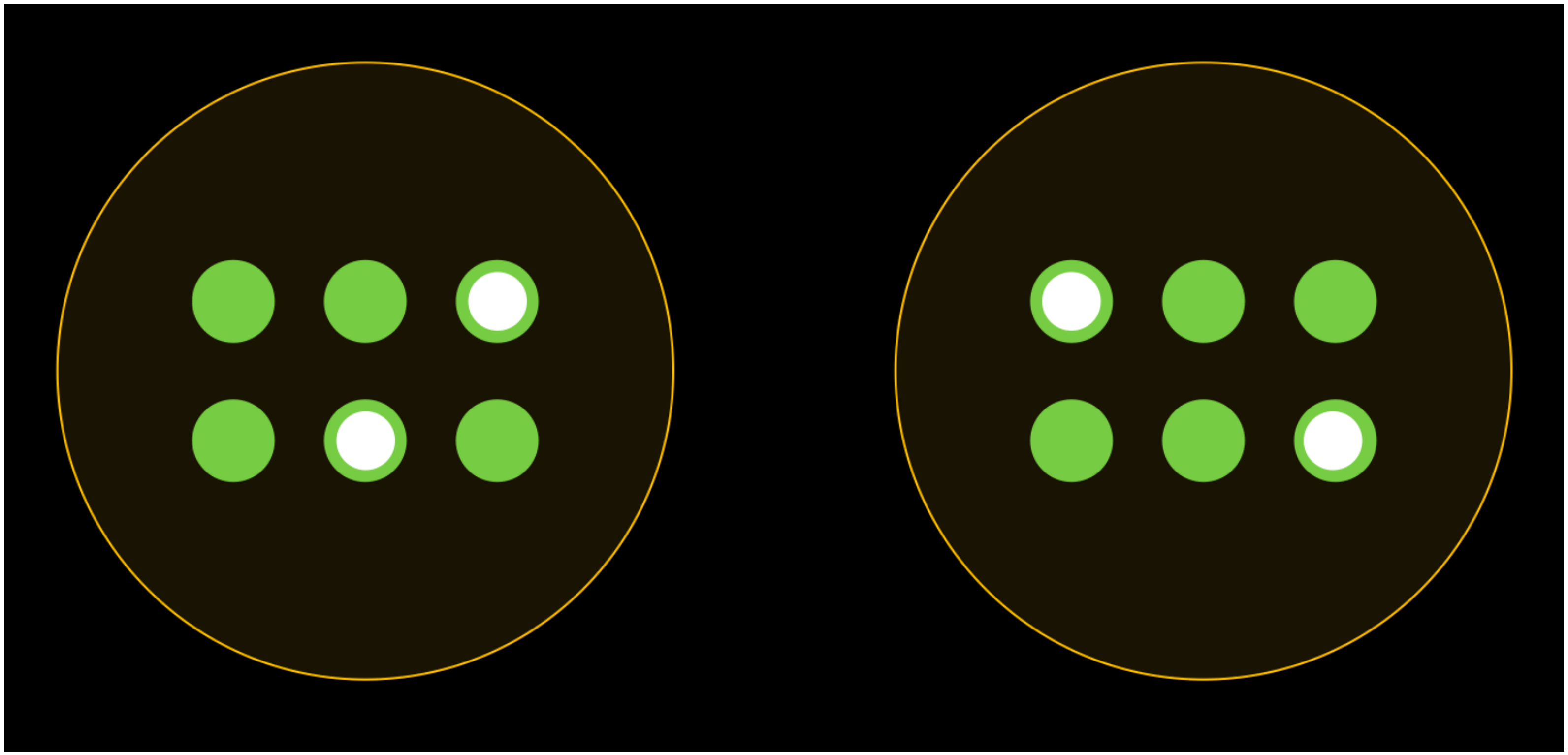
Locations



Locations and Requests



Presence Constraints



VRP with Location Constraints

- The problem
- Motivating applications
- Models
- Branch and Check and Price
- Experimental results

Motivating Applications

- Humanitarian and military logistics
  - number of possible landings in a base
  - number of parking spots available at a base
  - fuel available at an airport
- City logistics
  - parking at hubs
  - number of possible spots at transit centers

Service Constraints

- Service constraints
  - at most C_l vehicles can be served at a given location l simultaneously
- Cumulative constraint formulation

$$\textsc{Cumulative}(\{\text{serv}(i) : i \in \mathcal{R}_l\}, \{t(i) : i \in \mathcal{R}_l\}, \mathbf{1}, C_l)$$

starting dates

durations

resources

Presence Constraints

- Presence constraints
  - at most C_l vehicles can be present at a given location l simultaneously

- Cumulative constraint formulation

$$cumulative(\{arr(i) : i \in \Re_l\}, \{dep(i) - arr(i) : i \in \Re_l\}, 1, C_l)$$

arrival dates

presence times

resources

VRP with Location Constraints

- The problem
- Motivating applications
- Models
- Branch and Check and Price
- Experimental results

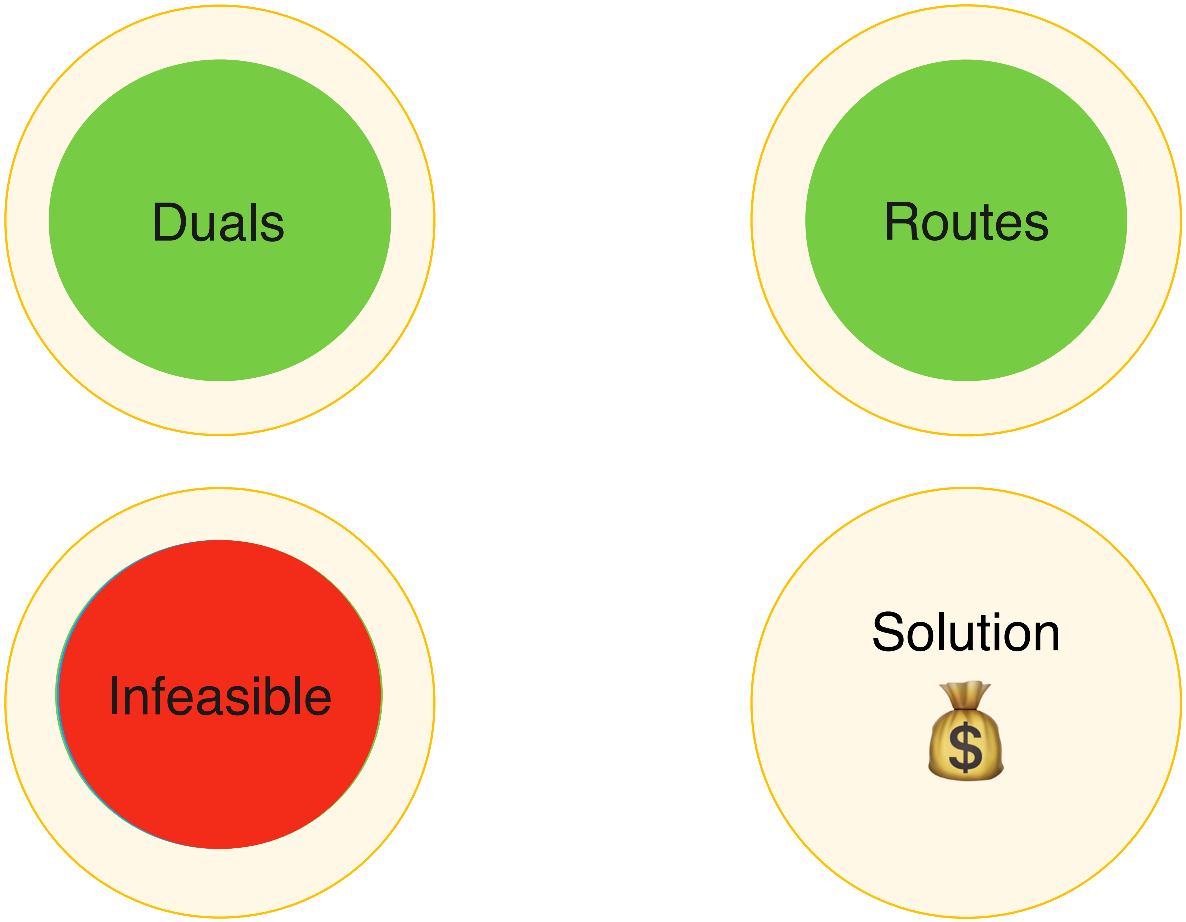
VRP with Location Constraints

- MIP
  - cumulative constraints implemented with logical constraints (no time-indexed formulation)
- CP
  - VRP + cumulative constraints
- Branch and Price and Check
  - branch and price
  - checking cumulative constraints with CP

VRP with Location Constraints

- The problem
- Motivating applications
- Cumulative Constraints
- Models
- Branch and Check and Price
- Experimental results

Branch and Price and Check



The Master Problem

r is a route

$$\min \sum_{r \in \Omega} c_r x_r$$

every pickup is covered by a single route

subject to

$$\sum_{r \in \Omega} a_{i,r} x_r = 1, \qquad \forall i \in \mathcal{P},$$

$$\qquad \forall B \in \mathcal{B},$$

$$x_r \in [0,1], \qquad \forall r \in \Omega.$$

The Pricing Problem

- Standard labeling algorithm
  - solving a resource-constrained shortest path with
    - time windows
    - pickup and delivery constraints
    - capacity constraints on the load of the vehicles
- Reduced costs

$$\bar{d}_{ij} = \begin{cases} d_{ij} - \pi_i + \sum_{B \in \mathcal{B}} 1_{Bij} \mu_B, & \forall i \in \mathcal{P}, j \in \mathcal{N}, \\ d_{ij} \qquad\quad + \sum_{B \in \mathcal{B}} 1_{Bij} \mu_B, & \forall i \in \mathcal{N} \setminus \mathcal{P}, j \in \mathcal{N}, \end{cases}$$

The Separation Problem

- The subproblem is heavily combinatorial
  - no simple Benders composition

$$
\begin{array}{ll}
\text{arr}(i) \leq \text{serv}(i), & \forall i \in \mathcal{R}, \\
\text{serv}(i) + t(i) \leq \text{dep}(i), & \forall i \in \mathcal{R}, \\
\text{arr}(i) = \text{serv}(i) = \text{dep}(i), & \forall i \in \mathcal{S} \cup \mathcal{E}, \\
\text{dep}(i) + d(i, \text{succ}(i)) = \text{arr}(\text{succ}(i)), & \forall i \in \mathcal{R} \cup \mathcal{S}, \\
\textsc{Cumulative}(\{\text{serv}(i) : i \in \mathcal{R}_l\}, \{t(i) : i \in \mathcal{R}_l\}, \mathbf{1}, C & \forall l \in \mathcal{L}.
\end{array}
$$

Separation Problem

- What does it do?

$$\mathrm{arr}(i) \leq \mathrm{serv}(i),$$
$$\mathrm{serv}(i) + t(i) \leq \mathrm{dep}(i),$$
$$\mathrm{arr}(i) = \mathrm{serv}(i) = \mathrm{dep}(i),$$
$$\mathrm{dep}(i) + d(i, \mathrm{succ}(i)) = \mathrm{arr}(\mathrm{succ}(i)),$$
$$\textsc{Cumulative}(\{\mathrm{serv}(i) : i \in \mathcal{R}_l\}, \{t(i) : i \in \mathcal{R}_l\}, \mathbf{1}, C_l),$$

- Determine
  - arrival and departure times for each vehicle
  - reasoning about scheduling and times

Separation Problem

- What does it do?
- Determine
  - arrival and departure times for each vehicle
  - reasoning about scheduling and times
- My inputs
  - are the routes for each vehicle

The Separation Problem

- How do we exclude incompatible routes?
  - what are the combinatorial Benders cuts?
- What is the main issue in excluding routes?

Combinatorial Benders Cuts

$$\sum_{r \in \Omega} B_r x_r \leq |B| - 1, \quad \forall B \in \mathcal{B},$$

Numbers of arcs in B for a route r

Arcs in the infeasible routes

Branching

- Consider a route

$$r = (i_1, i_2, \ldots, i_{n-1}, i_n)$$

- Branch on its prefixes $(i_1, i_2, \ldots, i_j)$

  – select all the arcs in the prefix
  – remove (i_j, i_{j+1})

VRP with Location Constraints

- The problem
- Motivating applications
- Models
- Branch and Check and Price
- Experimental results

Experimental Results

- Two Hours CP Time
  - (Gurobi as the MIP and LP solvers)

		MIP			CP		BPC		
P	C_l	Obj	Time	Gap	Obj	Time	Obj	Time	Gap
10	1	479	354	0.0 %	479	52	479	0	0.0 %
	2	479	1572	0.0 %	479	58	479	0	0.0 %
20	1	–	–	–	–	–	811	417	0.0 %
	2	929	–	66.1 %	–	–	771	0	0.0 %
	3	882	–	64.1 %	–	–	771	0	0.0 %
	4	–	–	–	–	–	771	0	0.0 %
	5	–	–	–	–	–	771	0	0.0 %

Experimental Results

		MIP			CP		BPC		
P	C_l	Obj	Time	Gap	Obj	Time	Obj	Time	Gap
60	1	–	–	–	–	0	–	–	–
	2	–	–	–	–	–	–	–	–
	3	–	–	–	–	–	1657	–	3.7 %
	4	–	–	–	–	–	1634	–	2.3 %
	5	–	–	–	–	–	1624	–	1.6 %
	6	–	–	–	–	–	1624	–	1.6 %
80	1	–	–	–	–	0	–	–	–
	2	–	–	–	–	–	–	–	–
	3	–	–	–	–	–	–	–	–
	4	–	–	–	–	–	2103	–	7.9 %
	5	–	–	–	–	–	1958	–	1.2 %
	6	–	–	–	–	–	1955	–	0.8 %
	7	–	–	–	–	–	1955	–	0.8 %
	8	–	–	–	–	–	1955	5278	0.0 %

Experimental Results

			Instance Set 1									Instance Set 2									Instance Set 3								
			TS	MIP			CP	BPC				TS	MIP			CP	BPC				TS	MIP			CP	BPC			
$\lvert\mathcal{L}\rvert$	$\lvert\mathcal{P}\rvert$	C_l	UB	UB	LB	Gap	UB	UB	LB	Gap	NG	UB	UB	LB	Gap	UB	UB	LB	Gap	NG	UB	UB	LB	Gap	UB	UB	LB	Gap	NG
8	15	1	–	–	45	–	204	**196**	196	0.0%	705	–	–	170	–	**359**	**359**	359	0.0%	634	–	–	24	–	150	**118**	118	0.0%	75
		2	–	–	47	–	212	**180**	180	0.0%	6	277	277	129	53.4%	**277**	**277**	277	0.0%	0	118	–	19	–	156	**118**	118	0.0%	0
		3	179	–	44	–	208	**179**	179	0.0%	0	277	277	159	42.6%	**277**	**277**	277	0.0%	0	118	–	21	–	143	**118**	118	0.0%	0
		4	179	208	46	77.9%	181	**179**	179	0.0%	0	277	277	204	26.4%	**277**	**277**	277	0.0%	0	118	–	21	–	175	**118**	118	0.0%	0
		5	179	–	47	–	181	**179**	179	0.0%	0	277	277	137	50.5%	**277**	**277**	277	0.0%	0	118	–	19	–	175	**118**	118	0.0%	0
		6	179	–	43	–	181	**179**	179	0.0%	0	277	277	210	24.2%	**277**	**277**	277	0.0%	0	118	133	23	82.7%	175	**118**	118	0.0%	0
		7	179	–	48	–	181	**179**	179	0.0%	0	277	277	230	17.0%	**277**	**277**	277	0.0%	0	118	–	20	–	175	**118**	118	0.0%	0
		8	179	191	45	76.4%	181	**179**	179	0.0%	0	277	277	206	25.6%	**277**	**277**	277	0.0%	0	118	–	24	–	175	**118**	118	0.0%	0
		9	179	207	44	78.7%	181	**179**	179	0.0%	0	277	277	194	30.0%	**277**	**277**	277	0.0%	0	118	–	18	–	175	**118**	118	0.0%	0
		10	179	–	47	–	181	**179**	179	0.0%	0	277	277	160	42.2%	**277**	**277**	277	0.0%	0	118	–	20	–	175	**118**	118	0.0%	0
		11	179	186	46	75.3%	181	**179**	179	0.0%	0	277	277	160	42.2%	**277**	**277**	277	0.0%	0	118	–	22	–	175	**118**	118	0.0%	0
		12	179	–	47	–	181	**179**	179	0.0%	0	277	277	159	42.6%	**277**	**277**	277	0.0%	0	118	–	16	–	175	**118**	118	0.0%	0
		13	179	–	47	–	181	**179**	179	0.0%	0	277	277	162	41.5%	**277**	**277**	277	0.0%	0	118	–	20	–	175	**118**	118	0.0%	0
		14	179	191	44	77.0%	181	**179**	179	0.0%	0	277	277	159	42.6%	**277**	**277**	277	0.0%	0	118	–	20	–	175	**118**	118	0.0%	0
		15	179	199	44	77.9%	181	**179**	179	0.0%	0	277	277	191	31.0%	**277**	**277**	277	0.0%	0	118	–	20	–	175	**118**	118	0.0%	0
11	15	1	–	245	100	59.2%	234	**223**	223	0.0%	20	–	–	374	–	×	×	–	–	74	–	124	42	66.1%	149	**114**	114	0.0%	290
		2	219	227	77	66.1%	243	**219**	219	0.0%	0	302	**302**	302	0.0%	**302**	**302**	302	0.0%	0	107	–	45	–	161	**107**	107	0.0%	0
		3	219	230	90	60.9%	272	**219**	219	0.0%	0	302	**302**	302	0.0%	**302**	**302**	302	0.0%	0	107	–	45	–	173	**107**	107	0.0%	0
		4	219	227	93	59.0%	272	**219**	219	0.0%	0	302	**302**	302	0.0%	415	**302**	302	0.0%	0	107	–	44	–	173	**107**	107	0.0%	0
		5	219	222	88	60.4%	272	**219**	219	0.0%	0	302	**302**	302	0.0%	415	**302**	302	0.0%	0	107	–	44	–	173	**107**	107	0.0%	0
		6	219	235	96	59.1%	272	**219**	219	0.0%	0	302	**302**	302	0.0%	415	**302**	302	0.0%	0	107	–	41	–	173	**107**	107	0.0%	0
		7	219	244	89	63.5%	272	**219**	219	0.0%	0	302	**302**	302	0.0%	415	**302**	302	0.0%	0	107	–	47	–	173	**107**	107	0.0%	0
		8	219	258	92	64.3%	272	**219**	219	0.0%	0	302	**302**	302	0.0%	415	**302**	302	0.0%	0	107	–	44	–	173	**107**	107	0.0%	0
		9	219	228	93	59.2%	272	**219**	219	0.0%	0	302	**302**	302	0.0%	415	**302**	302	0.0%	0	107	120	42	65.0%	173	**107**	107	0.0%	0
		10	219	228	93	59.2%	272	**219**	219	0.0%	0	302	**302**	302	0.0%	415	**302**	302	0.0%	0	107	–	43	–	173	**107**	107	0.0%	0
		11	219	228	93	59.2%	272	**219**	219	0.0%	0	302	**302**	302	0.0%	415	**302**	302	0.0%	0	107	–	44	–	173	**107**	107	0.0%	0
		12	219	228	93	59.2%	272	**219**	219	0.0%	0	302	**302**	302	0.0%	415	**302**	302	0.0%	0	107	–	43	–	173	**107**	107	0.0%	0
		13	219	228	92	59.6%	272	**219**	219	0.0%	0	302	**302**	302	0.0%	415	**302**	302	0.0%	0	107	143	42	70.6%	173	**107**	107	0.0%	0
		14	219	228	92	59.6%	272	**219**	219	0.0%	0	302	**302**	302	0.0%	415	**302**	302	0.0%	0	107	143	42	70.6%	173	**107**	107	0.0%	0
		15	219	219	102	53.4%	272	**219**	219	0.0%	0	302	**302**	302	0.0%	415	**302**	302	0.0%	0	107	–	43	–	173	**107**	107	0.0%	0

Table 4.5: Solutions to the instances with service resources. The upper bound (UB), lower bound (LB), optimality gap (Gap) and the number of nogoods (NG) are reported for each solver when available. The symbol × denotes a proof of infeasibility and values in bold indicate a proof of optimality.

Experimental Results

			Instance Set 1									Instance Set 2									Instance Set 3								
			TS	MIP			CP	BPC				TS	MIP			CP	BPC				TS	MIP			CP	BPC			
$\lvert\mathcal{L}\rvert$	$\lvert\mathcal{P}\rvert$	C_l	UB	UB	LB	Gap	UB	UB	LB	Gap	NG	UB	UB	LB	Gap	UB	UB	LB	Gap	NG	UB	UB	LB	Gap	UB	UB	LB	Gap	NG
8	80	1	–	–	0	–	×	–	746	–	3,171	–	–	0	–	×	–	1,176	–	28,124	–	–	0	–	×	–	390	–	2,865
		2	–	–	0	–	×	–	746	–	3,169	–	–	0	–	×	–	1,176	–	27,082	–	–	0	–	–	–	390	–	17
		3	–	–	0	–	–	–	746	–	13,897	–	–	0	–	×	–	1,176	–	28,660	–	–	0	–	–	–	391	–	2,730
		4	–	–	0	–	–	845	755	10.7%	2,971	–	–	0	–	–	–	1,176	–	28,040	393	–	0	–	–	**393**	393	0.0%	111
		5	794	–	0	–	–	799	755	5.5%	1,903	–	–	0	–	–	1,370	1,182	13.7%	20,805	393	–	0	–	–	**393**	393	0.0%	0
		6	794	–	0	–	–	795	761	4.3%	149	1,195	–	0	–	–	1,195	1,182	1.1%	2,565	393	–	0	–	–	**393**	393	0.0%	0
		7	794	–	0	–	–	794	762	4.0%	0	1,195	–	0	–	–	1,194	1,184	0.8%	1,729	393	–	0	–	–	**393**	393	0.0%	0
		8	794	–	0	–	–	794	762	4.0%	0	1,195	–	0	–	–	1,195	1,184	0.9%	2,242	393	–	0	–	–	**393**	393	0.0%	0
		9	794	–	0	–	–	794	762	4.0%	0	1,195	–	0	–	–	1,195	1,186	0.8%	0	393	–	0	–	–	**393**	393	0.0%	0
		10	794	–	0	–	–	794	762	4.0%	0	1,195	–	0	–	–	1,195	1,186	0.8%	0	393	–	0	–	–	**393**	393	0.0%	0
		11	794	–	0	–	–	794	762	4.0%	0	1,195	–	0	–	–	1,195	1,186	0.8%	0	393	–	0	–	–	**393**	393	0.0%	0
		12	794	–	0	–	–	794	762	4.0%	0	1,195	–	0	–	–	1,195	1,186	0.8%	0	393	–	0	–	–	**393**	393	0.0%	0
		13	794	–	0	–	–	794	762	4.0%	0	1,195	–	0	–	–	1,195	1,186	0.8%	0	393	–	0	–	–	**393**	393	0.0%	0
		14	794	–	0	–	–	794	762	4.0%	0	1,195	–	0	–	–	1,195	1,186	0.8%	0	393	–	0	–	–	**393**	393	0.0%	0
		15	794	–	0	–	–	793	763	3.8%	0	1,195	–	0	–	–	1,195	1,186	0.8%	0	393	–	0	–	–	**393**	393	0.0%	0
11	80	1	–	–	4	–	×	–	876	–	8,763	–	–	0	–	×	–	1,260	–	24,022	–	–	0	–	×	–	411	–	10,388
		2	–	–	4	–	×	–	872	–	12,877	–	–	0	–	×	–	1,260	–	25,102	–	–	0	–	–	–	407	–	12
		3	–	–	4	–	–	–	875	–	9,282	–	–	0	–	×	–	1,260	–	24,733	–	–	0	–	–	–	410	–	780
		4	–	–	4	–	–	–	881	–	914	–	–	0	–	–	–	1,260	–	25,974	413	–	0	–	–	413	411	0.5%	184
		5	883	–	4	–	–	**883**	883	0.0%	32	–	–	0	–	–	1,340	1,260	6.0%	11,902	413	–	0	–	–	413	412	0.2%	0
		6	883	–	4	–	–	**883**	883	0.0%	0	–	–	0	–	–	1,319	1,266	4.0%	5,011	413	–	0	–	–	413	412	0.2%	0
		7	883	–	4	–	–	**883**	883	0.0%	0	–	–	0	–	–	1,312	1,269	3.3%	2,083	413	–	0	–	–	413	412	0.2%	0
		8	883	–	4	–	–	**883**	883	0.0%	0	1,280	–	0	–	–	1,280	1,270	0.8%	2,165	413	–	0	–	–	413	412	0.2%	0
		9	883	–	4	–	–	**883**	883	0.0%	0	1,280	–	0	–	–	1,280	1,272	0.6%	0	413	–	0	–	–	413	412	0.2%	0
		10	883	–	4	–	–	**883**	883	0.0%	0	1,280	–	0	–	–	1,280	1,272	0.6%	0	413	–	0	–	–	413	412	0.2%	0
		11	883	–	4	–	–	**883**	883	0.0%	0	1,280	–	0	–	–	1,280	1,272	0.6%	0	413	–	0	–	–	413	412	0.2%	0
		12	883	–	4	–	–	**883**	883	0.0%	0	1,280	–	0	–	–	1,280	1,272	0.6%	0	413	–	0	–	–	413	412	0.2%	0
		13	883	–	4	–	–	**883**	883	0.0%	0	1,280	–	0	–	–	1,280	1,272	0.6%	0	413	–	0	–	–	413	412	0.2%	0
		14	883	–	4	–	–	**883**	883	0.0%	0	1,280	–	0	–	–	1,280	1,272	0.6%	0	413	–	0	–	–	413	411	0.5%	0
		15	883	–	4	–	–	**883**	883	0.0%	0	1,280	–	0	–	–	1,280	1,272	0.6%	0	413	–	0	–	–	413	412	0.2%	0

Table 4.5: Solutions to the instances with service resources. The upper bound (UB), lower bound (LB), optimality gap (Gap) and the number of nogoods (NG) are reported for each solver when available. The symbol × denotes a proof of infeasibility and values in bold indicate a proof of optimality. (Continued on next page)

Experimental Results

			Instance Set 1									Instance Set 2									Instance Set 3								
			TS	MIP			CP	BPC				TS	MIP			CP	BPC				TS	MIP			CP	BPC			
$\|\mathcal{L}\|$	$\|\mathcal{P}\|$	C_l	UB	UB	LB	Gap	UB	UB	LB	Gap	NG	UB	UB	LB	Gap	UB	UB	LB	Gap	NG	UB	UB	LB	Gap	UB	UB	LB	Gap	NG
8	40	1	–	–	14	–	×	–	426	–	18,661	–	–	0	–	×	–	625	–	36,911	–	–	0	–	×	–	234	–	12,487
		2	–	–	0	–	–	–	427	–	8,693	–	–	0	–	×	–	625	–	27,030	–	–	0	–	–	–	235	–	3,647
		3	431	–	0	–	–	**431**	431	0.0%	6	–	–	0	–	–	645	625	3.1%	16,350	–	–	0	–	–	266	235	11.7%	2,353
		4	431	–	0	–	–	**431**	431	0.0%	0	634	–	0	–	–	**634**	634	0.0%	0	–	–	0	–	–	239	238	0.4%	882
		5	431	–	0	–	–	**431**	431	0.0%	0	634	–	0	–	–	**634**	634	0.0%	0	239	–	0	–	–	**239**	239	0.0%	0
		6	431	–	0	–	–	**431**	431	0.0%	0	634	–	0	–	–	**634**	634	0.0%	0	239	–	0	–	–	**239**	239	0.0%	0
		7	431	–	0	–	–	**431**	431	0.0%	0	634	–	0	–	–	**634**	634	0.0%	0	239	–	0	–	–	**239**	239	0.0%	0
		8	431	–	0	–	–	**431**	431	0.0%	0	634	–	0	–	–	**634**	634	0.0%	0	239	–	0	–	–	**239**	239	0.0%	0
		9	431	–	0	–	–	**431**	431	0.0%	0	634	–	0	–	–	**634**	634	0.0%	0	239	–	0	–	–	**239**	239	0.0%	0
		10	431	–	0	–	–	**431**	431	0.0%	0	634	–	0	–	–	**634**	634	0.0%	0	239	–	0	–	–	**239**	239	0.0%	0
		11	431	–	0	–	–	**431**	431	0.0%	0	634	–	0	–	–	**634**	634	0.0%	0	239	–	0	–	–	**239**	239	0.0%	0
		12	431	–	0	–	–	**431**	431	0.0%	0	634	–	0	–	–	**634**	634	0.0%	0	239	–	0	–	–	**239**	239	0.0%	0
		13	431	–	0	–	–	**431**	431	0.0%	0	634	–	0	–	–	**634**	634	0.0%	0	239	–	0	–	–	**239**	239	0.0%	0
		14	431	–	0	–	–	**431**	431	0.0%	0	634	–	0	–	–	**634**	634	0.0%	0	239	–	0	–	–	**239**	239	0.0%	0
		15	431	–	0	–	–	**431**	431	0.0%	0	634	–	0	–	–	**634**	634	0.0%	0	239	–	0	–	–	**239**	239	0.0%	0
11	40	1	–	–	14	–	×	–	489	–	12,301	–	–	50	–	×	–	682	–	32,708	–	–	4	–	×	–	236	–	10,368
		2	–	–	14	–	–	556	495	11.0%	6,782	–	–	28	–	×	–	682	–	34,062	–	–	0	–	–	**238**	238	0.0%	250
		3	499	–	14	–	–	**499**	499	0.0%	0	–	–	42	–	–	772	682	11.7%	24,610	238	–	0	–	–	**238**	238	0.0%	1
		4	499	–	14	–	–	**499**	499	0.0%	0	–	–	28	–	–	716	688	3.9%	20,026	238	–	0	–	–	**238**	238	0.0%	0
		5	499	–	4	–	–	**499**	499	0.0%	0	696	–	40	–	–	**696**	696	0.0%	30	238	–	0	–	–	**238**	238	0.0%	0
		6	499	–	5	–	–	**499**	499	0.0%	0	696	–	28	–	–	**696**	696	0.0%	0	238	–	2	–	–	**238**	238	0.0%	0
		7	499	–	14	–	–	**499**	499	0.0%	0	696	–	49	–	–	**696**	696	0.0%	0	238	–	0	–	–	**238**	238	0.0%	0
		8	499	–	14	–	–	**499**	499	0.0%	0	696	–	34	–	–	**696**	696	0.0%	0	238	–	0	–	–	**238**	238	0.0%	0
		9	499	–	14	–	–	**499**	499	0.0%	0	696	–	56	–	–	**696**	696	0.0%	0	238	–	0	–	–	**238**	238	0.0%	0
		10	499	–	14	–	–	**499**	499	0.0%	0	696	–	36	–	–	**696**	696	0.0%	0	238	–	0	–	–	**238**	238	0.0%	0
		11	499	–	14	–	–	**499**	499	0.0%	0	696	–	44	–	–	**696**	696	0.0%	0	238	–	0	–	–	**238**	238	0.0%	0
		12	499	–	14	–	–	**499**	499	0.0%	0	696	–	45	–	–	**696**	696	0.0%	0	238	–	0	–	–	**238**	238	0.0%	0
		13	499	–	14	–	–	**499**	499	0.0%	0	696	–	42	–	–	**696**	696	0.0%	0	238	–	0	–	–	**238**	238	0.0%	0
		14	499	–	14	–	–	**499**	499	0.0%	0	696	–	40	–	–	**696**	696	0.0%	0	238	–	0	–	–	**238**	238	0.0%	0
		15	499	–	14	–	–	**499**	499	0.0%	0	696	–	41	–	–	**696**	696	0.0%	0	238	–	0	–	–	**238**	238	0.0%	0

Table 4.6: Solutions to the instances with presence resources. The upper bound (UB), lower bound (LB), optimality gap (Gap) and the number of nogoods (NG) are reported for each solver when available. The symbol × denotes a proof of infeasibility and values in bold indicate a proof of optimality. (Continued on next page)

[redacted]

Computational Optimization

Branch and Price and Check

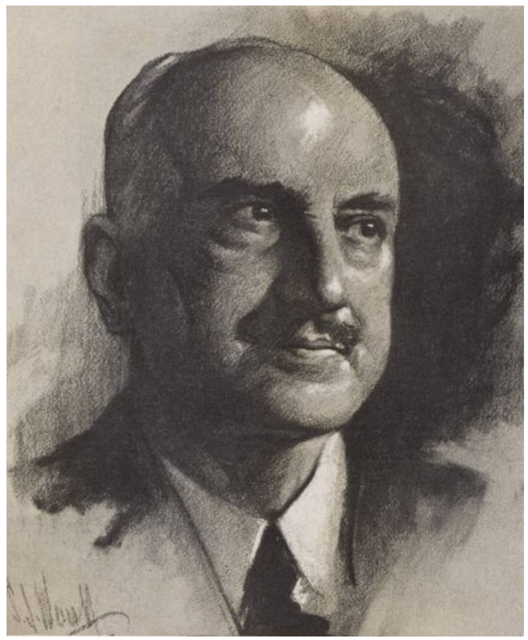
Pascal Van Hentenryck

Solving the Separability Problem

- Learning-based constraint programming
  - CP + learning from failures
- Analyze each failure
  - conflict analysis

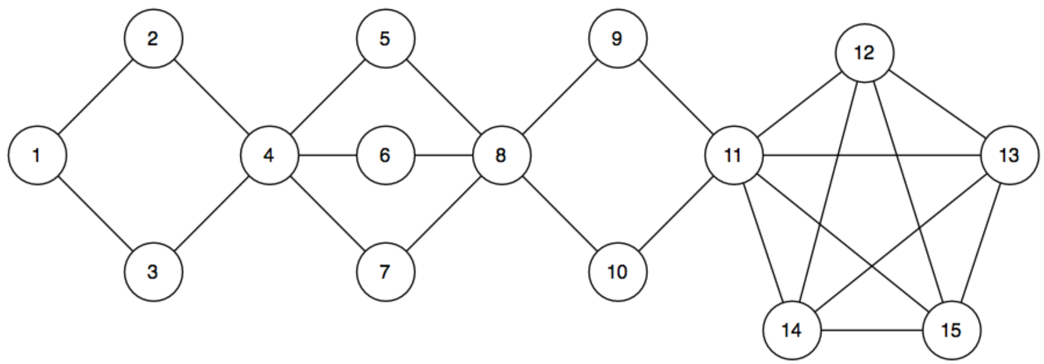
Learning

- Those who forget the past are doomed to repeat it



How Much Search is Repeated?

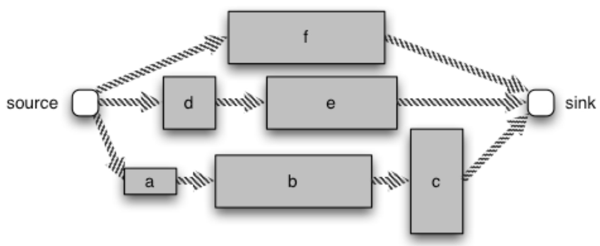
- Color the following graph with 4 colors

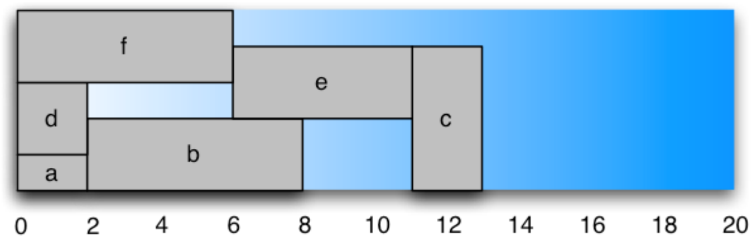


- Traditional search: 462672 failures
- Constraint Programming with Learning
  - 18 failures

How Much Search is Repeated?

- Resource Constrained Project Scheduling
  - BL instance (20 tasks)





- Input order: 934,535 failures
  - With learning: 931 failures
- Smallest start time order: 296,567 failures
  - With learning: 551 failures
- Activity-based search: > 2,000,000 failures
  - With learning: 1144 failures

[redacted]

Conflict-Based Learning in CP

- Can we learn from failures?
- A long history (from at least 1986)
  - but it has not really worked until recently

SAT Solvers

- Very simple models
  - clauses in CNF
- Very simple propagation
  - unit propagation
- Activity-Based Search (VSIDS)
  - branch on variables appearing in failures the most
- Conflict-based learning
  - learn a new clause (1UIP) when encountering a failure

Conflict-Based Learning in CP

- Keep the model structure
  - avoid clause/variable blasting
- Ability to program the search
  - exploiting the problem structure
- Exploit the structure for conflict analysis
  - build an inference graph
  - each constraint should “explain” its inferences
- Learn strong constraints (aka no-goods)
  - nogood = disjunction of constraints
    - semantically redundant, i.e., implied by the model
    - compensates weaknesses of the propagation
    - captures “global” information

Inference Graph

- What does pruning do in CP?
  - $x = v$, $x \neq v$, $x \geq v$, $x \leq v$ (in its simplest form)
    - domain events
  - denote them as $[x = v]$, $[x \neq v]$, $[x \geq v]$, $[x \leq v]$
- The graph captures the CP Inferences
  - e.g. $[x \leq 2]$ and $x \geq y$ implies that $[y \leq 2]$
  - inference: $[x \leq 2] \rightarrow [y \leq 2]$

Inference Graph

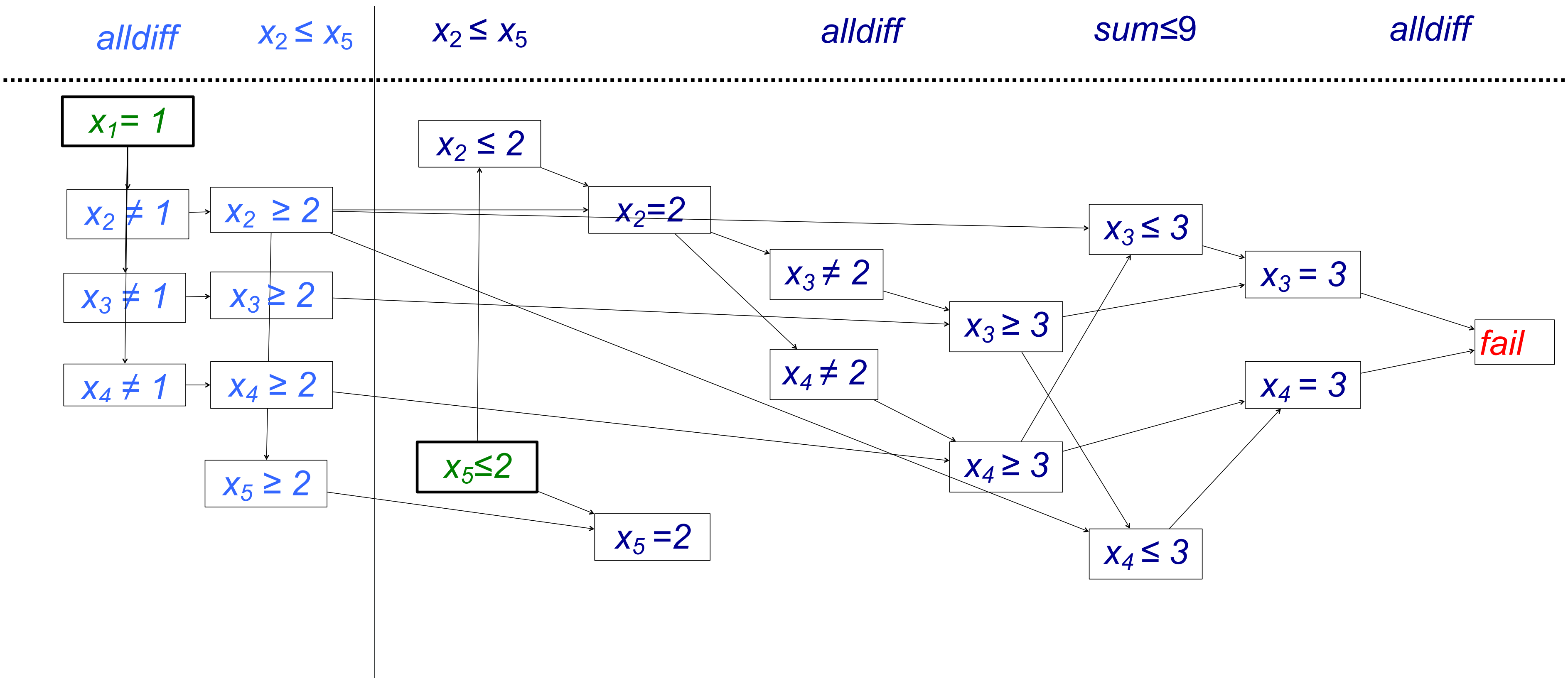
- What does pruning do in CP?
  - $x = v$, $x \neq v$, $x \geq v$, $x \leq v$ (in its simplest form)
- Inference rules
  - alldifferent([x[1],x[2],x[3],x[4]);
  - Setting $x[1] = 1$ we generate new inferences
    - $x[2] \neq 1$, $x[3] \neq 1$, $x[4] \neq 1$
  - Inferences
    - $[x[1] = 1] \rightarrow [x[2] \neq 1]$,
    - $[x[1] = 1] \rightarrow [x[3] \neq 1]$,
    - $[x[1] = 1] \rightarrow [x[4] \neq 1]$

Constraint Propagation

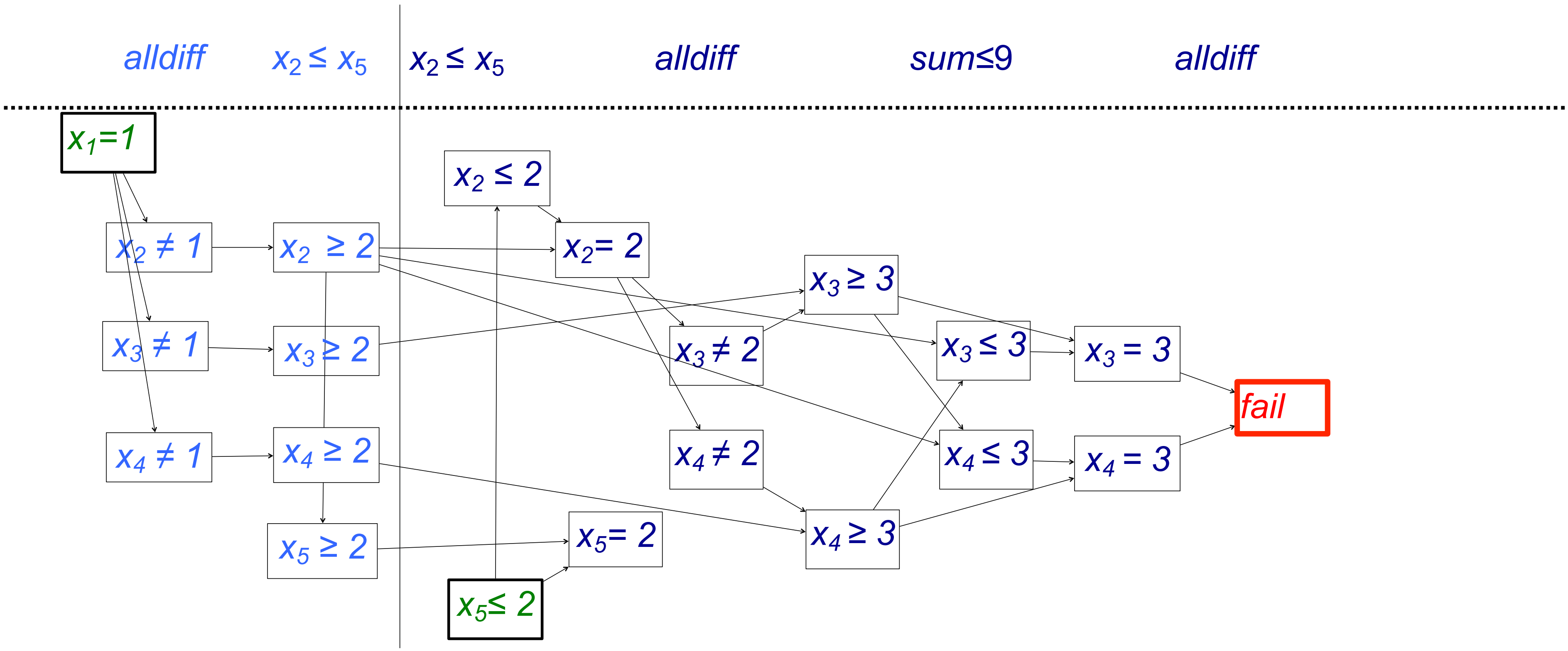
```
dvar int x[1..5] in 1..4;
constraints {
   allDifferent(all(i in 1..4) x[i]);
   x[2] <= x[5];
   sum(i in 1..4) x[i] <= 9;
}
```

	x[1] =1	alldiff	x[2]≤ x[5]	x[5] >2	x[2] ≤ x[5]	alldiff	sum≤9	alldiff
x[1]	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*
x[2]	*1..4*	*2..4*	*2..4*	*2..4*	*2*	*2*	*2*	*2*
x[3]	*1..4*	*2..4*	*2..4*	*2..4*	*2..4*	*3..4*	*3*	✖
x[4]	*1..4*	*2..4*	*2..4*	*2..4*	*2..4*	*3..4*	*3*	✖
x[5]	*1..4*	*1..4*	*2..4*	*3..4*	*2*	*2*	*2*	*2*

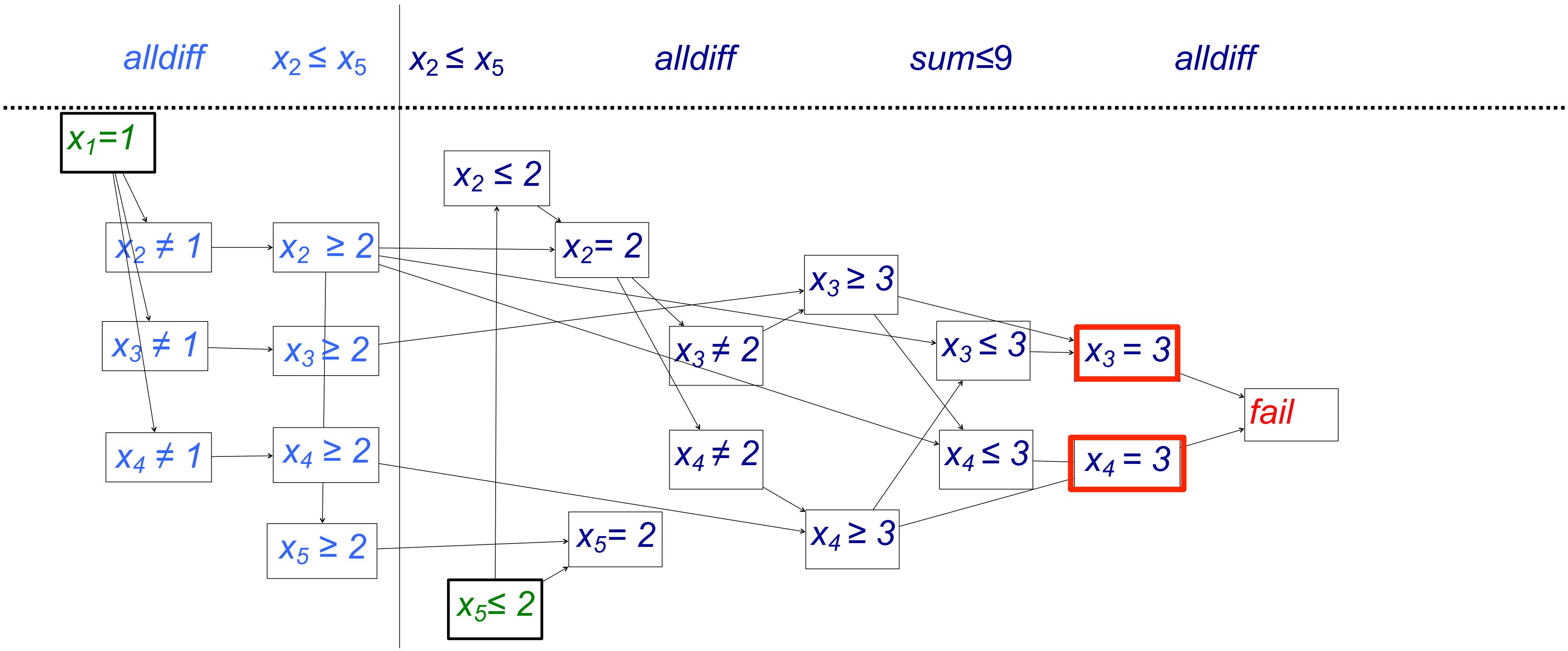
Inference Graph



Nogood Learning

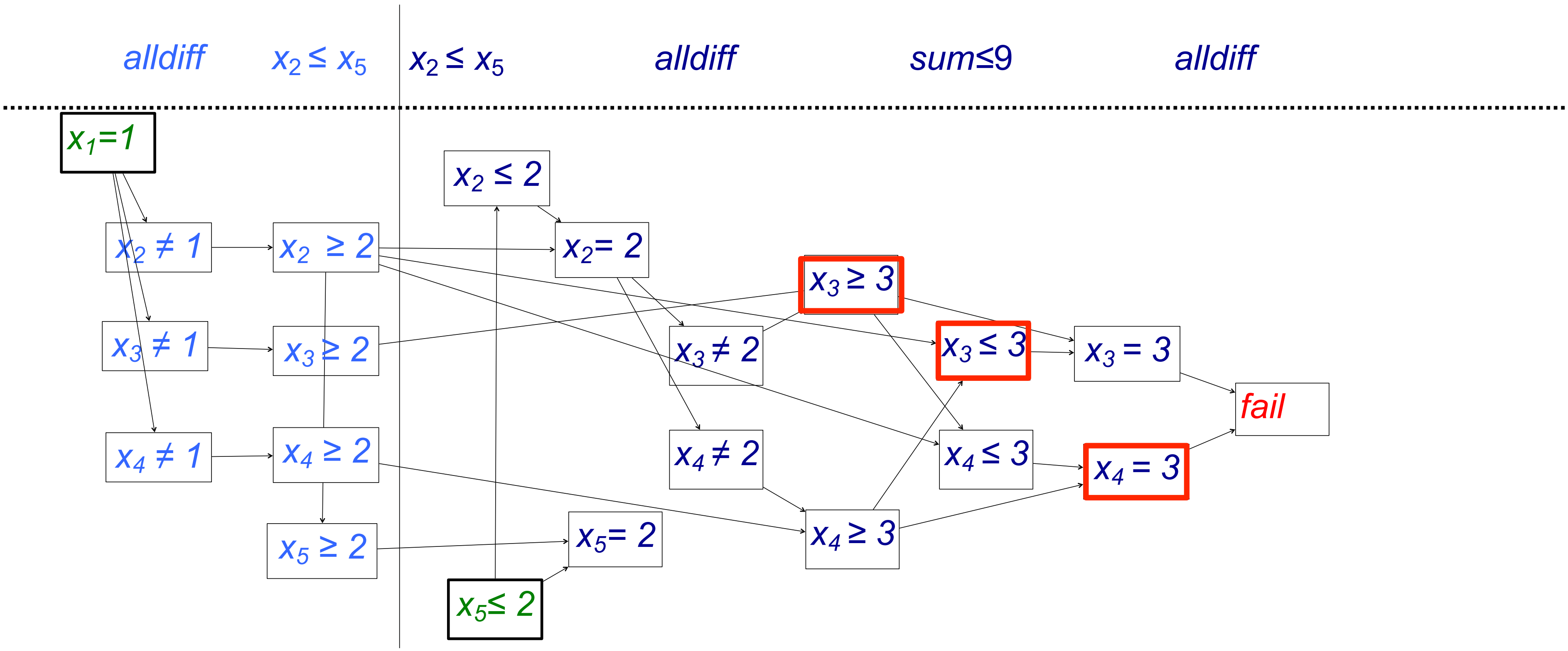


Nogood Learning



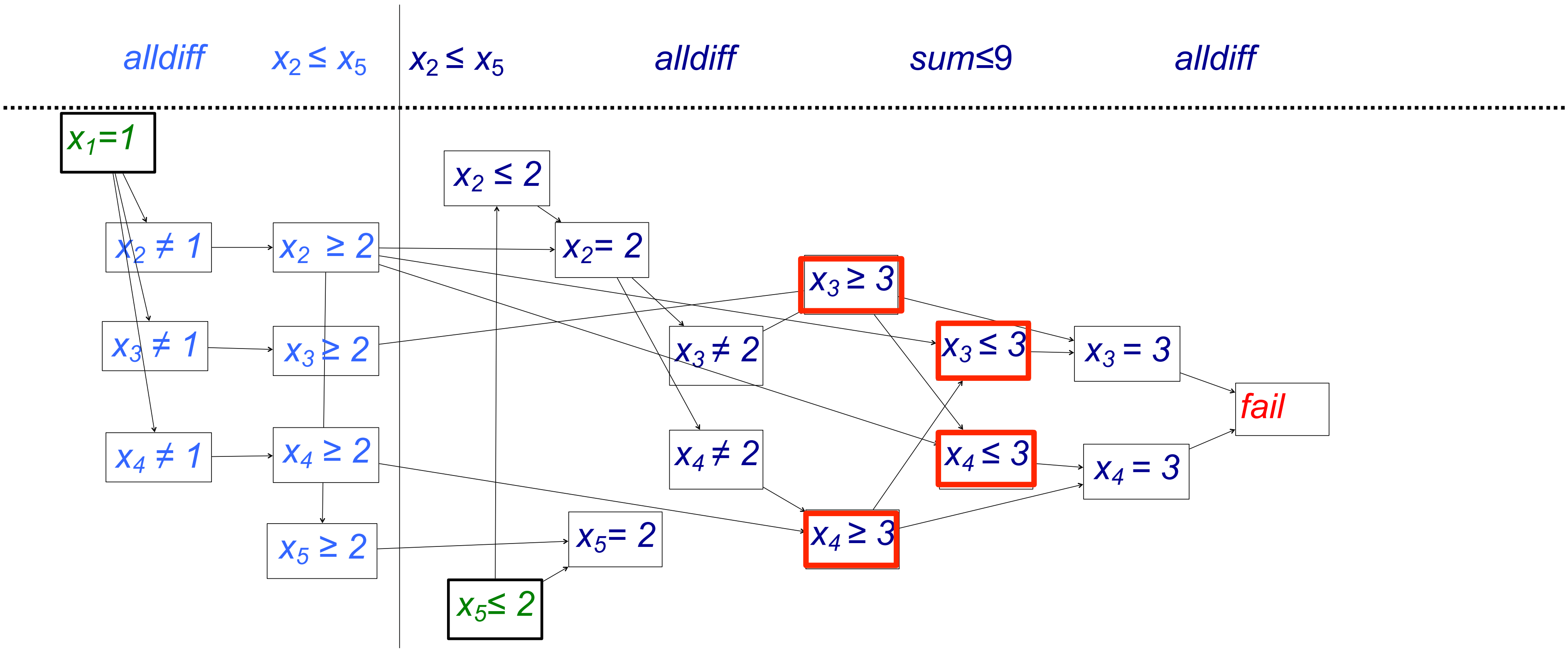
$x_3=3 \wedge x_4=3 \rightarrow$ *false*

Nogood Learning



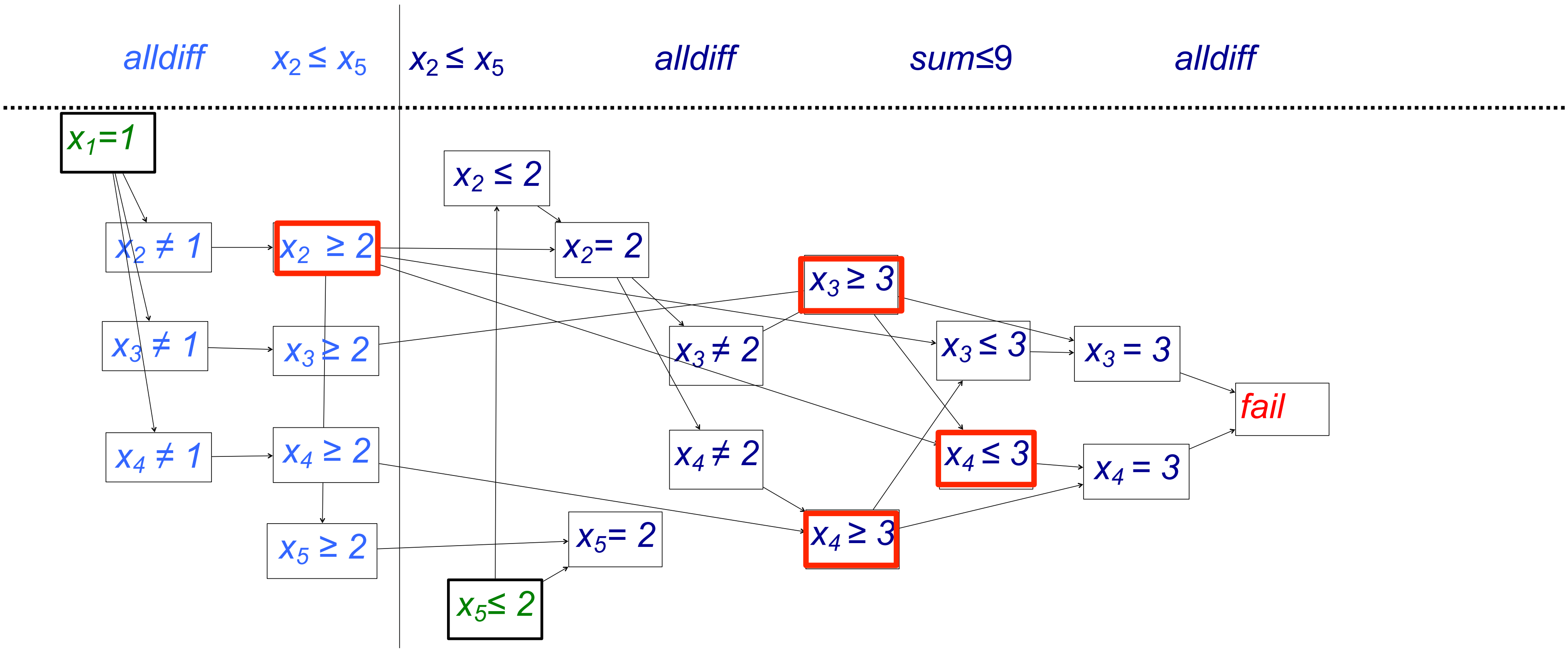
$x_3 \geq 3 \land x_3 \leq 3 \land x_4 = 3 \rightarrow$ *false*

Nogood Learning



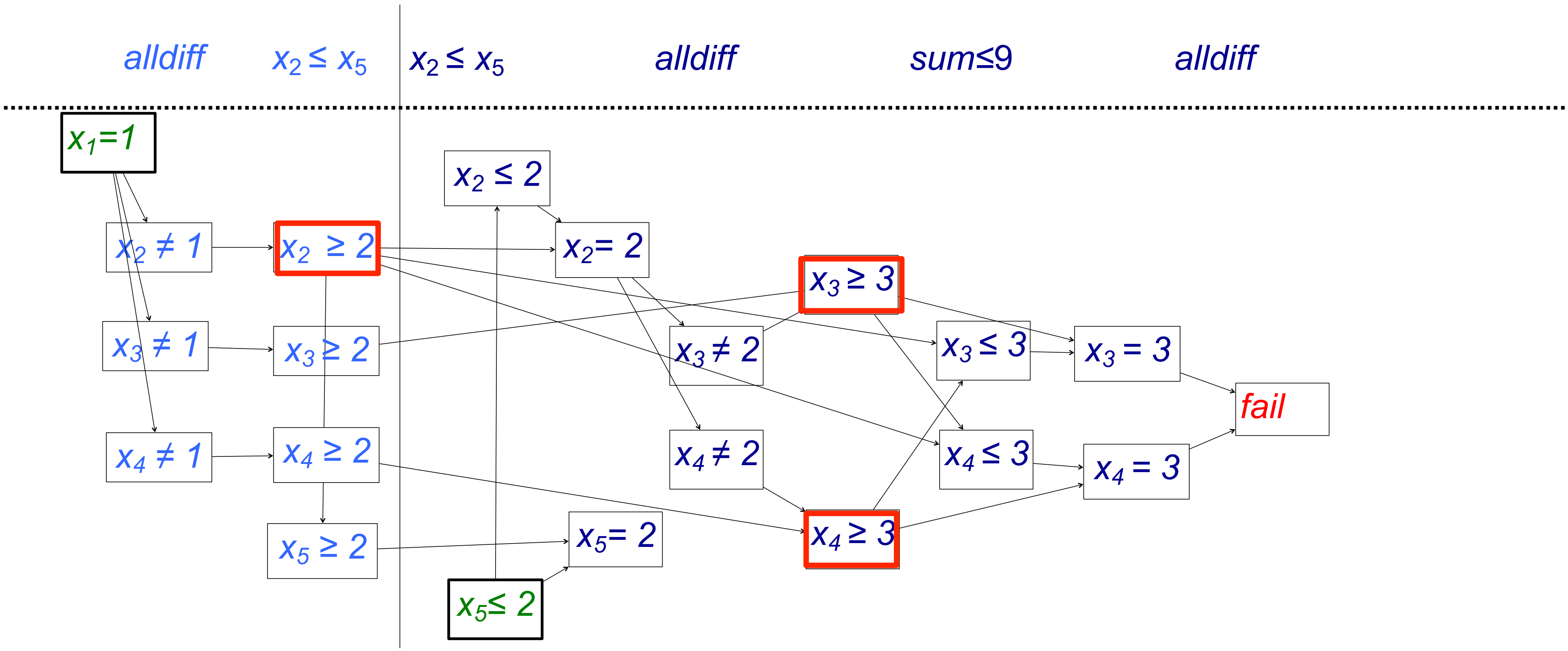
$\{x_3 \geq 3, x_4 \geq 3, x_3 \leq 3, x_4 \leq 3\} \rightarrow$ *false*

Nogood Learning



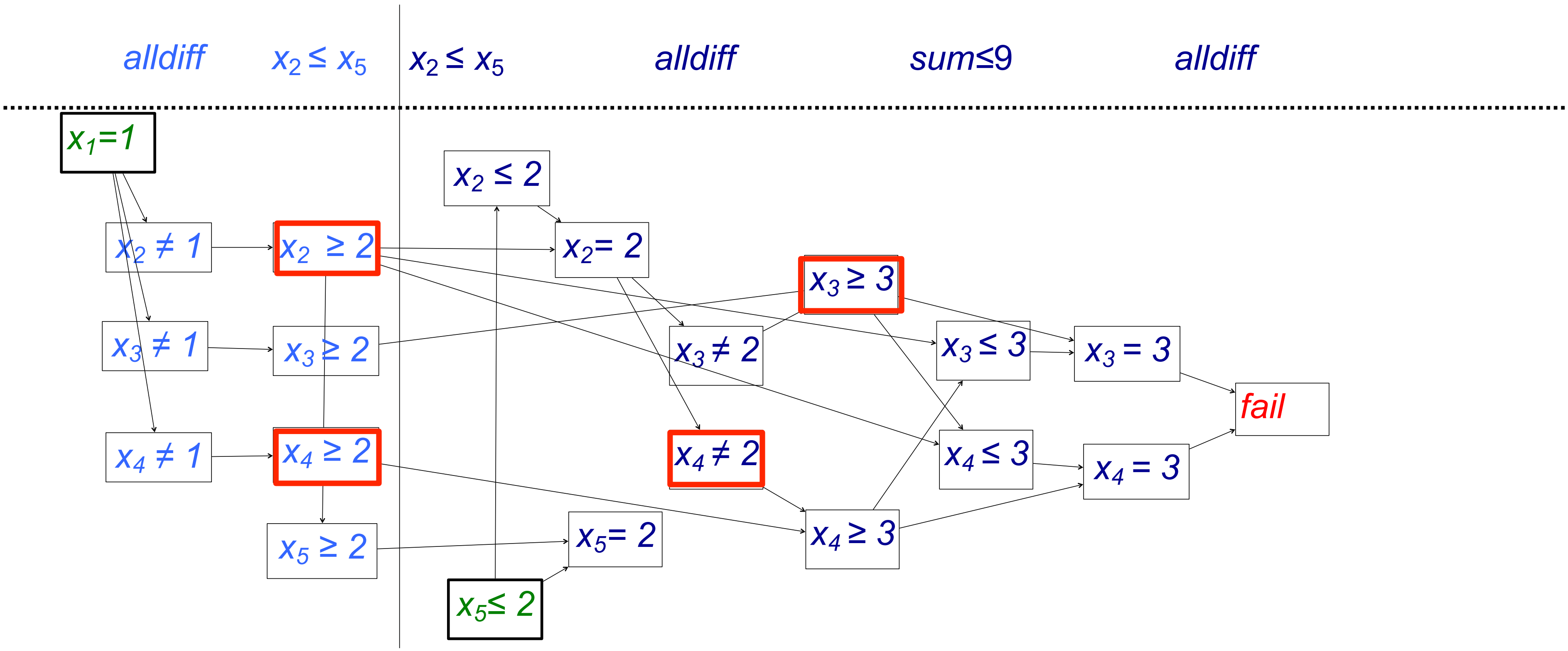
$\{x_2 \geq 2, x_3 \geq 3, x_4 \geq 3, x_3 \leq 3\} \rightarrow$ *false*

Nogood Learning



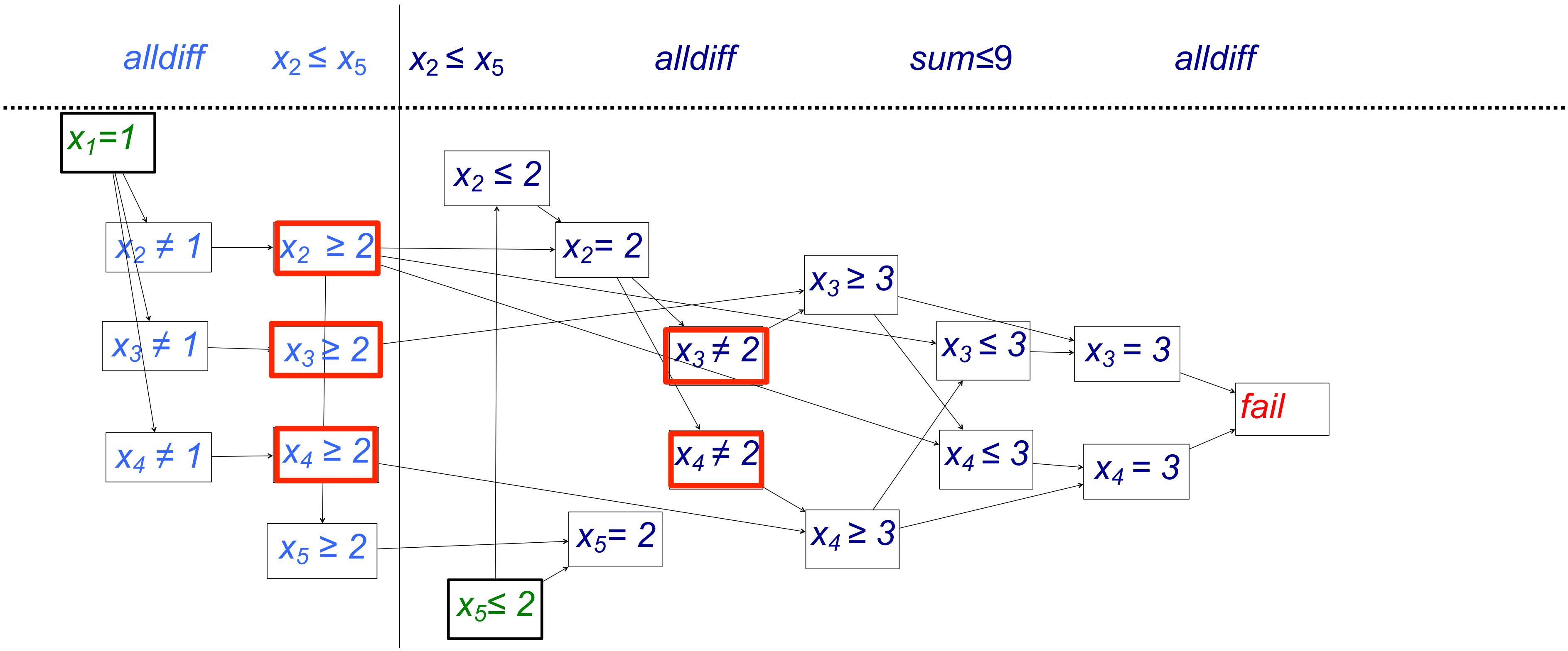
$\{x_2 \geq 2, x_3 \geq 3, x_4 \geq 3\} \rightarrow$ *false*

Nogood Learning



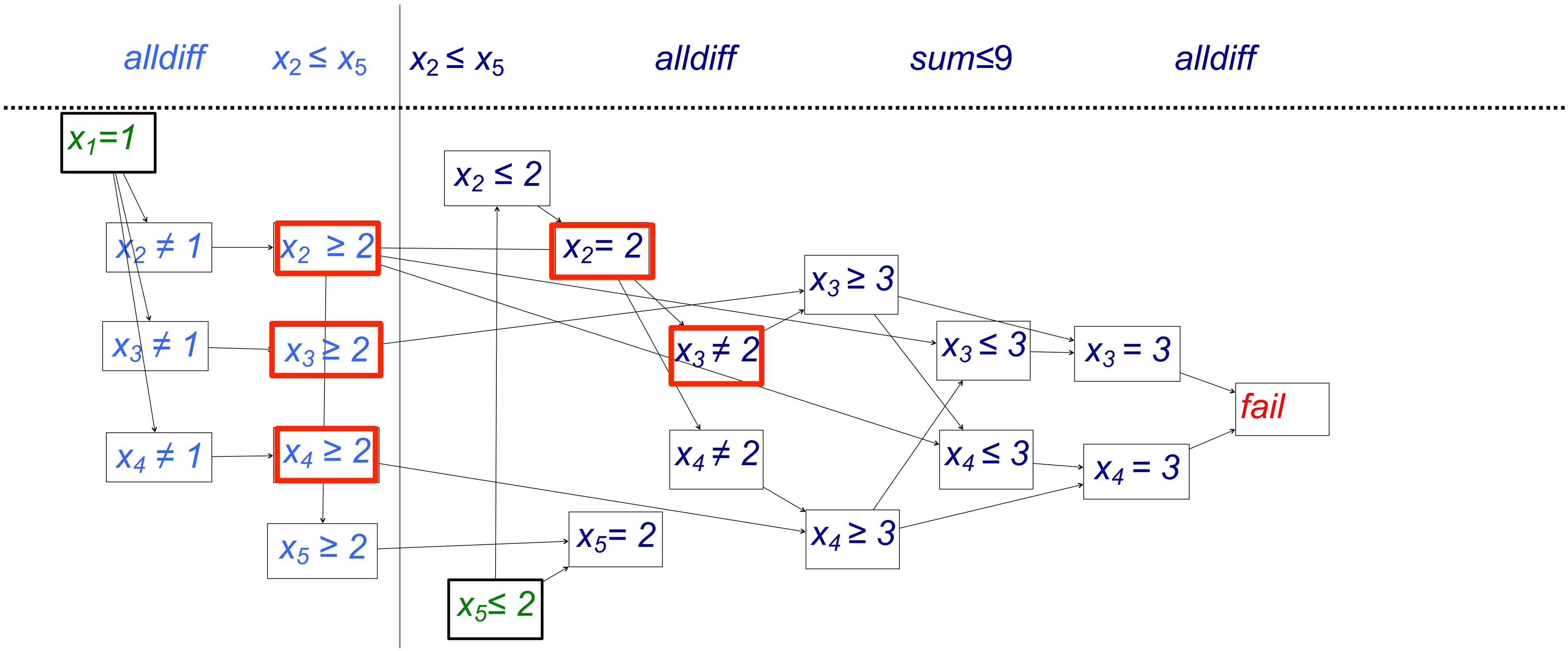
$\{x_2 \geq 2, x_4 \geq 2, x_4 \neq 2, x_3 \geq 3\} \rightarrow$ *false*

Nogood Learning



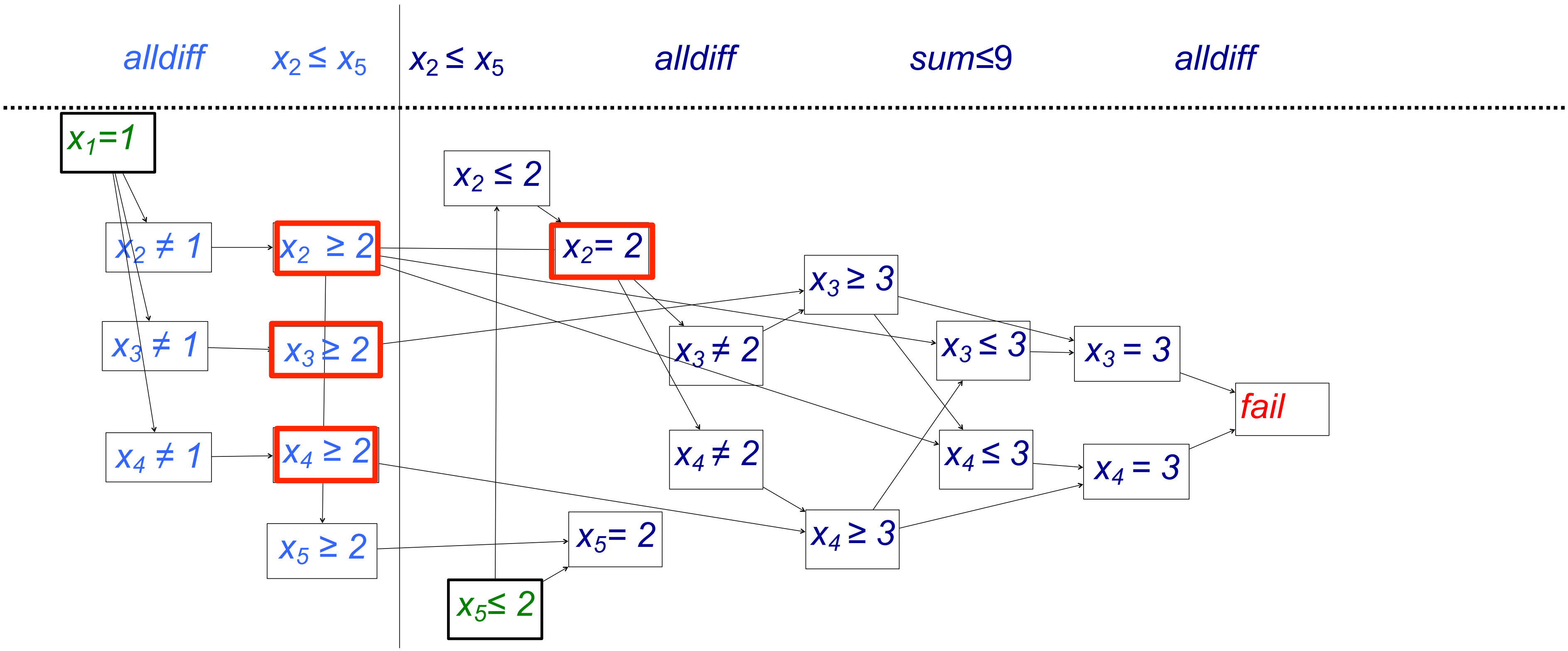
$\{x_2 \geq 2, x_3 \geq 2, x_4 \geq 2, x_3 \neq 2, x_4 \neq 2\} \rightarrow$ *false*

Nogood Learning



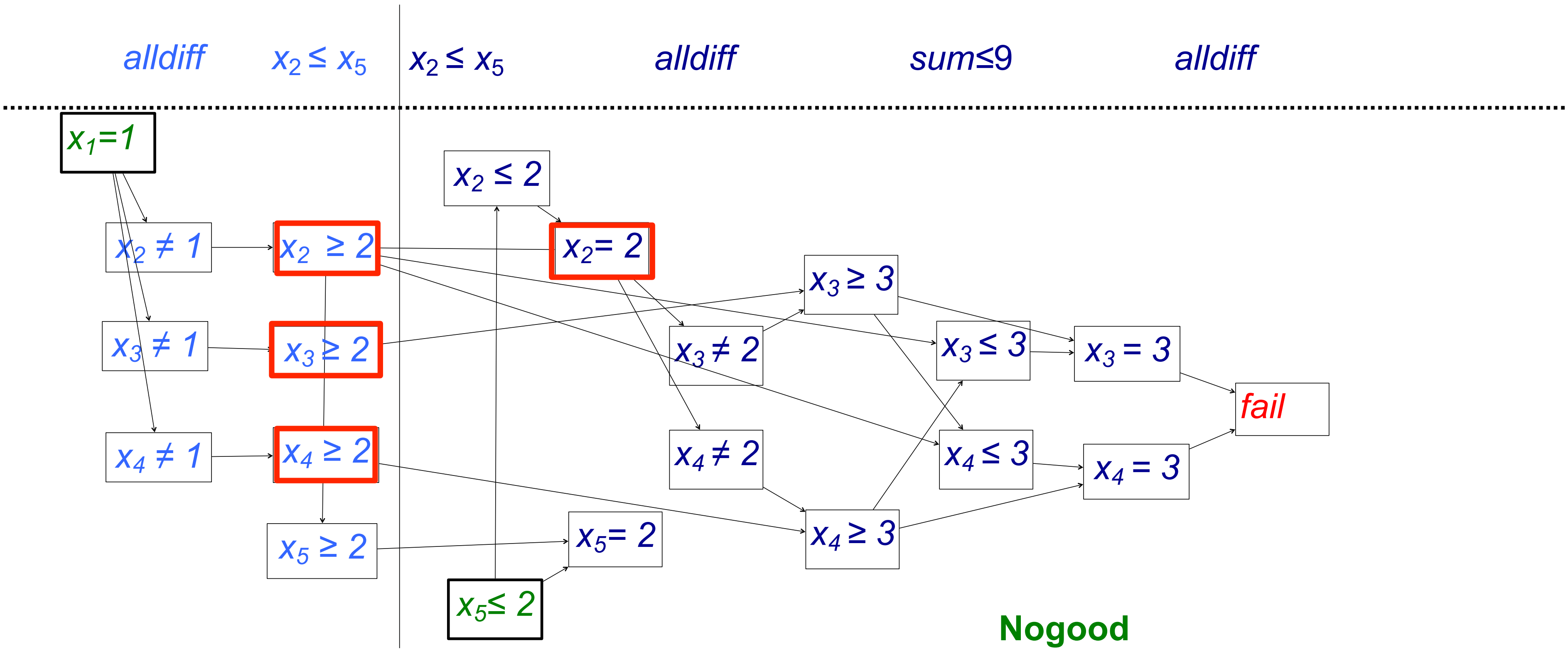
$\{x_2 \geq 2, x_3 \geq 2, x_4 \geq 2, x_2 = 2, x_3 \neq 2\} \rightarrow$ *false*

Nogood Learning



$\{x_2 \geq 2, x_3 \geq 2, x_4 \geq 2, x_2 = 2\} \rightarrow$ *false*

Nogood Learning



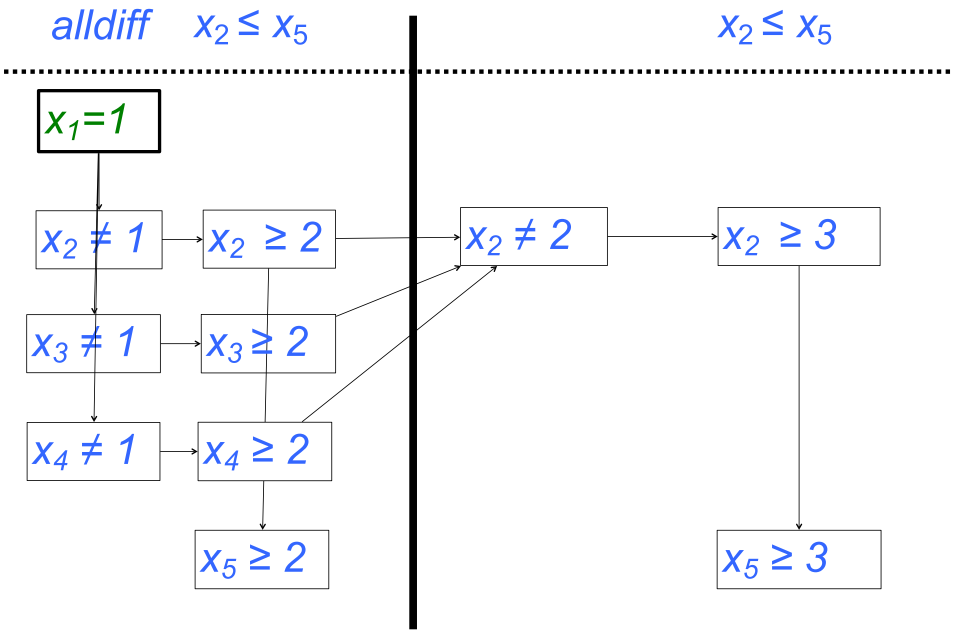
$\{x_2 \geq 2, x_3 \geq 2, x_4 \geq 2, x_2 = 2\} \rightarrow$ *false*

Nogood

$x_3 \leq 1 \vee x_4 \leq 1 \vee x_2 \neq 2$

Backtracking

- Backtrack to second last level in nogood



$\{x_2 \geq 2, x_3 \geq 2, x_4 \geq 2, x_2 = 2\} \rightarrow$ *false*

- Nogood will propagate
- Note stronger domain than usual backtracking
  - $D(x_2) = \{3..4\}$

Nogood

$x_3 \leq 1 \vee x_4 \leq 1 \vee x_2 \neq 2$

Conflict-Based Learning in CP

- Inference graph
  - can be built implicitly, on demand
- Each constraint should explain its inferences
  - either during propagation
    - forward explanations
    - record inferences
  - or during conflict analysis
    - backward explanations
    - ask constraints for inferences

Perspectives

- Trends in combinatorial optimization
  - logical Benders, and branch & check
  - generalizes Benders decomposition with combinatorial subproblem
- Branch and price and check
  - generalizes the idea to branch and price
- Beautiful synergies between
  - mathematical and constraint programming


"That's all folks!"

Computational Optimization

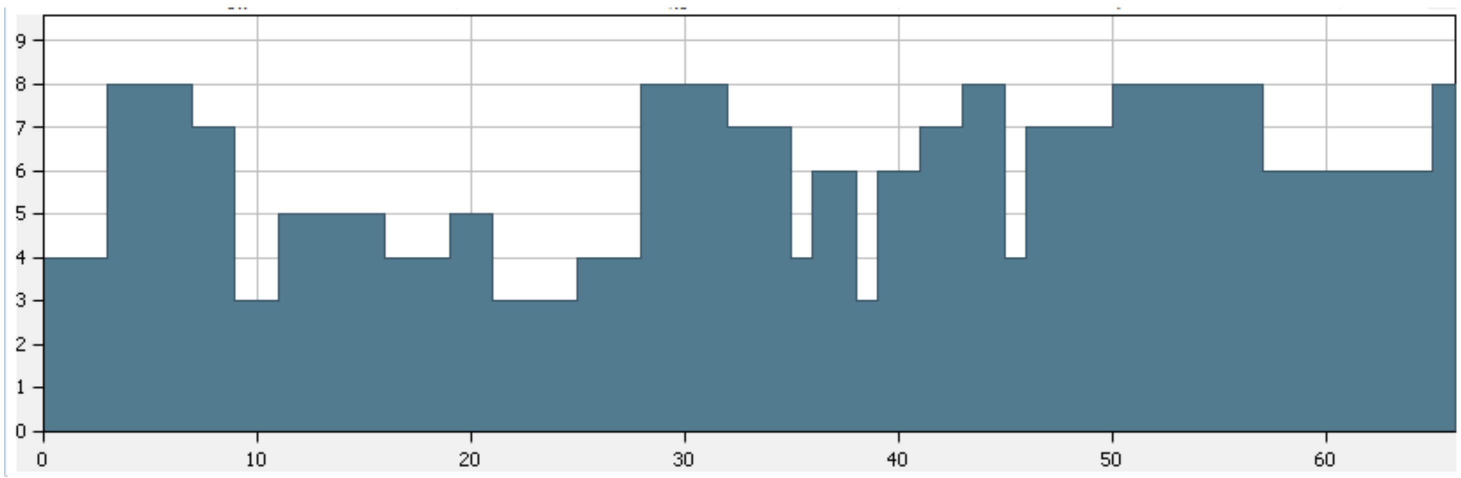
Logical-Based Benders Decomposition

Pascal Van Hentenryck

Logic Benders Decomposition

- Limitation of Benders Decomposition
  - the subproblem must be convex
- Logical Benders Decomposition
  - the subproblems are discrete optimization problems
  - the Benders cuts are obtained from the structure of the subproblems
  - dedicated to the problem at hand

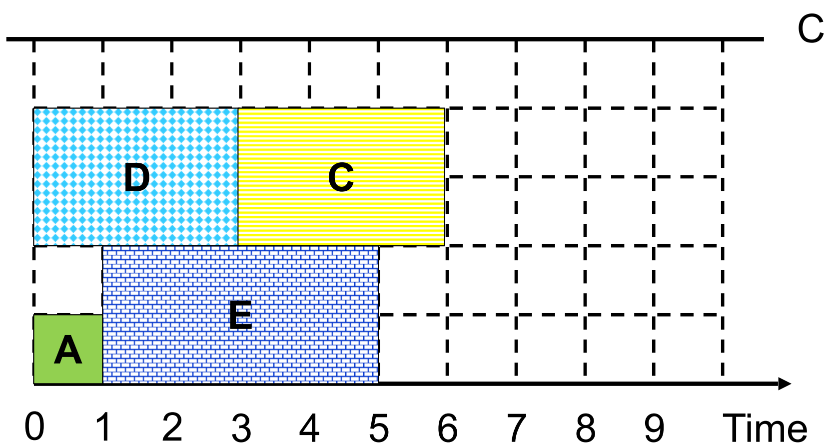
Cumulative Constraints



Cumulative Constraints

- Given a task set T and a resource with a capacity of C then an assignment S of the start times is a solution iff

$$\forall t \in [t, t_{\max}) : \sum_{i \in T : S(i) \leq t < S(i) + p_i} c_i \leq C$$



Machine Scheduling

- Given
  - a set I of tasks with time windows in which they can execute
  - a set of machines M with capacities on which the tasks can execute
    - cumulative resources
  - a cost c_{im} for executing task i on machine m
  - each task i has earlier and latest release dates $[r_i,e_i]$
  - each task i has a duration d_i
- minimize the total cost
  - satisfying the time windows and capacity constraints

Machine Scheduling

- Decision variables
  - $x_{i,m}$: 1 if task i is assigned to machine m
  - a_i: starting time of task i
- Constraints
  - each task is assigned to a single machine
  - cumulative constraints
  - time windows

Machine Scheduling

- The optimization problem

$$
\begin{array}{llll}
\min\limits_{i\in I, m\in M} & c_{i,m}\ x_{i,m} & & \\
\text{subject to} & & & \\
& \sum_{m\in M} x_{i,m} = 1 & & (i \in I) \\
& \mathit{cumulative}(& \{i \in T \ \mid \ x_{i,m} = 1\}, & \\
& & \{a_i | i \in T\}, & \\
& & \{d_i | i \in T\}, & \\
& & C_m) & (m \in M) \\
& a_i \in [r_i, e_i] & & (i \in I) \\
& x_{im} \in \{0, 1\} & & (i \in I, m \in M)
\end{array}
$$

Logical Benders Decomposition

- The subproblems
  - are single-machine feasibility problems
  - can be solved by constraint programming
- The master problem is simply

$$
\begin{array}{lll}
\min\limits_{i \in I, m \in M} & c_{i,m}\ x_{i,m} & \\
\text{subject to} & & \\
& \sum_{m \in M} x_{i,m} = 1 & (i \in I) \\
& \text{Benders Cuts} & \\
& x_{i,m} \in \{0,1\} & (i \in I, m \in M)
\end{array}
$$

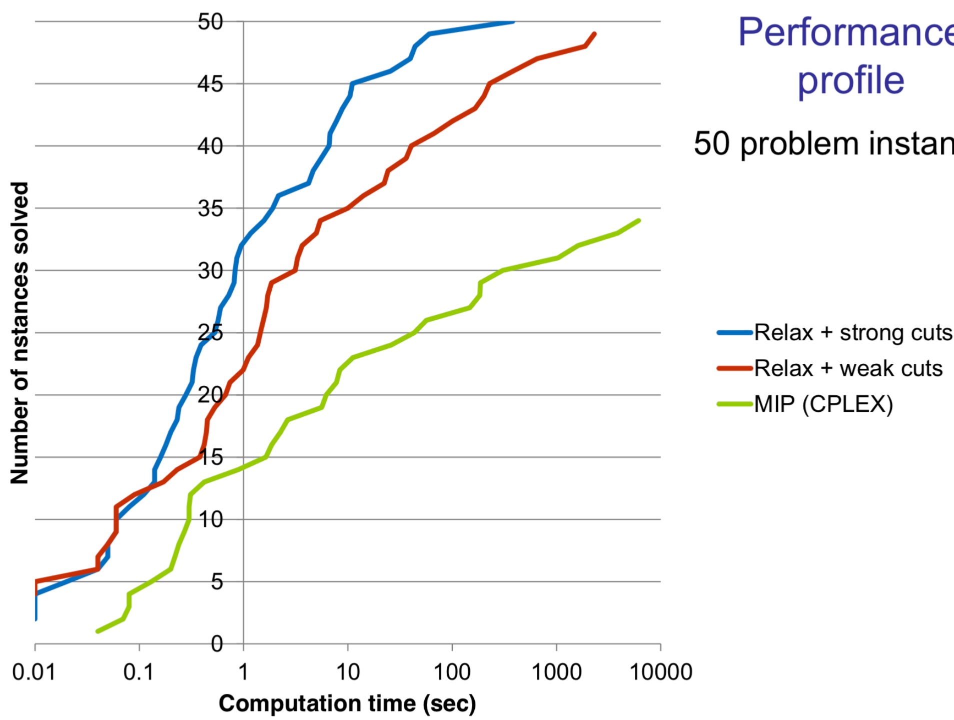
Logical Benders Decomposition

- What are the Benders cuts?
  - assume that a cumulative subproblem is infeasible
  - what can we deduce?
- The logical Benders cuts for an assignment $\bar{x}$
  - ensures that this assignment is never produced again

$$\sum_{k \in \{i | \bar{x}_{i,m} = 1\}} x_{k,m} \leq |\{i \mid \bar{x}_{i,m} = 1\}| - 1$$

  - can be strengthened by identifying the tasks that are creating the failures

Logical Benders Decomposition



Performance profile

50 problem instances

Branch and Check

- Benders decomposition
  - solves the master problem at optimality before considering the subproblem
- Branch and check
  - solves the subproblems at various nodes of the search tree
- Most natural use
  - generates a Benders cut each time a leaf is reached

Branch and Check

- Very similar to branch and cut
- Not the same as branch and cut
  - In branch and cut, the cuts contain unfixed variables.
  - In branch and check, the cuts contain fixed variables.
- When to use?
  - When master problem is the bottleneck.
  - The master problem is solved only once, with a growing constraint set.

[redacted]

Computational Optimization

Advanced Topics in Column Generation: Part I

Pascal Van Hentenryck

Motivation

Bounding the Column Generation

- Finding a lower bound
  - why?

Bounding the Column Generation

- Finding a lower bound
  - column generation only produces a lower bound — on completion —
- Column generation may take a long time to converge
- a tight lower bound may be available sooner
  - and often does!

Motivation

- Lower bounds for column generation
  - A lower bound based on Lagrangian relaxation
  - Farley's Bound

Notations

- The Master Problem (MP)

$$\begin{array}{ll} \min & c\,x \\ \text{subject to} & \\ & Ax \geq b \\ & x_j \geq 0 \end{array}$$

- Assumption: an upper bound on the sum of the variables is available

$$\kappa \geq \sum_{i=1}^{n} x_i$$

A Lagrangian Relaxation Lower Bound

- Searching for a lower bound to the master problem
  - LB(MP) $\leq$ O(MP)
- Current optimum to the Restricted Master Problem (RMP)
  - O(RMP)
- Smallest reduced cost to the RMP:
  - O(PP) where PP is the pricing problem
- The lower bound:

$$O(RMP) + \kappa \, O(PP) \leq O(MP)$$

Proof (by Lagrangian relaxation)

$$\begin{array}{ll} \min & c\,x \\ \text{subject to} & \\ & Ax \geq b \\ & x_j \geq 0 \end{array}$$

$$\begin{array}{ll} \min & c\,x + \Pi(b - Ax) \\ \text{subject to} & \\ & x_j \geq 0 \\ & (\text{with } \Pi \geq 0) \end{array}$$

Proof (by Lagrangian relaxation)

$$\begin{array}{ll} \min & c\,x \\ \text{subject to} & \\ & Ax \geq b \\ & x_j \geq 0 \end{array} \qquad \begin{array}{ll} \min & c\,x + \Pi(b - Ax) \\ \text{subject to} & \\ & x_j \geq 0 \\ & (\text{with } \Pi \geq 0) \end{array}$$

- At a given iteration, the RMP has the reduced costs

$$cx = \Pi b + (c - \Pi A)x$$

- where

$$\Pi = c_B A_B^{-1}$$

- The proof uses these simplex multipliers (dual values)

Proof (by Lagrangian relaxation)

$$cx^* \geq cx^* + \Pi(b - Ax^*)$$

$$cx^* \geq \Pi b + (c - \Pi A)x^*$$

reduced costs in the current iteration

$$cx^* \geq \Pi b + \bar{c}x^*$$

$$cx^* \geq \Pi b + \sum_{i=1}^{n} \bar{c}_i x_i^*$$

$$cx^* \geq \Pi b + \sum_{i=1}^{n} \bar{c}_{min} x_i^*$$

smallest reduced cost: It is non-positive

$$cx^* \geq \Pi b + \bar{c}_{min} \sum_{i=1}^{n} x_i^*$$

this is bounded by assumption

$$cx^* \geq \Pi b + \bar{c}_{min} \kappa$$

O(RMP)

Motivation

- Lower bounds with column generation
  - A lower bound based on Lagrangian relaxation
  - Farley's Bound

More Lower Bounds

- Farley's Bound
  - amazingly easy and beautiful

Primal / Dual

$$
\begin{array}{ll}
\min & c\,x \\
\text{subject to} & \\
& Ax \geq b \\
& x_j \geq 0 \\
& (c \geq 0)
\end{array}
\qquad\qquad
\begin{array}{ll}
\max & b^T\,y \\
\text{subject to} & \\
& A^T y \leq c \\
& y_j \geq 0
\end{array}
$$

- Assume that we have primal and dual optimal solutions to the primal problem with a subset of the columns

$$\bar{x} \text{ and } \bar{y}$$

The Farley's Bound

Reduced cost for column j: $c_j - \bar{y}^T A_j$

$$Let \ \ \lambda_j = \bar{y}^T A_j$$

$$Let \ \ \frac{c_k}{\lambda_k} = \min_{j:\lambda_j > 0} \frac{c_j}{\lambda_j}$$

Theorem (Farley)

$$c\bar{x}\frac{c_k}{\lambda_k} \text{ is a lower bound to the primal problem}$$

Intuition

- Consider $c_j = 1$
- Then

$$\frac{1}{\lambda_k} = \min_{j:\lambda_j > 0} \frac{1}{\lambda_j} = \max_{j:\lambda_j > 0} \lambda_j$$

- and we are selecting the smallest reduced cost

$$c_j - \bar{y}^T A_j$$

- Observe that the lower bound

$$\frac{c\bar{x}}{\lambda_k}$$

- converges towards 1.

Proof of Farley's Bound

By duality, $c\bar{x}\frac{c_k}{\lambda_k} = b^T\bar{y}\frac{c_k}{\lambda_k}$. By duality, it suffices to show that $\bar{y}\frac{c_k}{\lambda_k}$ is a feasible to the dual. Since $\frac{c_k}{\lambda_k} \geq 0$ by definition, $\bar{y}\frac{c_k}{\lambda_k} \geq 0$.

Consider constraint j, i.e.,

$$(A_j)^T\bar{y}\frac{c_k}{\lambda_k} \leq c_j \equiv \lambda_j\frac{c_k}{\lambda_k} \leq c_j$$

If $\lambda_j \leq 0$, then the result follows since $c_j \geq 0$. Otherwise,

$$\lambda_j\frac{c_k}{\lambda_k} \leq c_j \equiv \frac{c_k}{\lambda_k} \leq \frac{c_j}{\lambda_j}$$

which follows by definition of $\frac{c_k}{\lambda_k}$.

Limitations of Farley's Bound

- How to compute Farley's bound?
  - from the minimum reduced cost
- Limitations of Farley's bound
  - We need to know the cost vector c
    - not applicable when the pricing subproblem computes c_j
  - It is hard to find the best ratio
    - when the cost vector c does not have some special properties

[redacted]

Computational Optimization

Advanced Topics in Column Generation: Part II

Pascal Van Hentenryck

Pascal Van Hentenryck

Stabilized Column Generation

- Slow convergence of column generation
  - e.g., degeneracy of P

(P)

$$
\begin{array}{ll}
\min & c\ x \\
\text{subject to} & \\
& Ax = b \\
& x \geq 0
\end{array}
$$

(D)

$$
\begin{array}{ll}
\max & \pi b \\
\text{subject to} & \\
& \pi A \leq c
\end{array}
$$

Column Generation

- Restricted Master Problem

$$\begin{array}{lll} \min & c^k \ x^k \\ \text{subject to} & \\ & A^k x^k = b \\ & x^k \geq 0 \end{array} \qquad \begin{array}{ll} \max & \pi^k b \\ \text{subject to} & \\ & \pi^k A^k \leq c^k \end{array}$$

Column Generation

- What is the pricing problem doing?

$$
\begin{array}{ll}
\min & c^k \; x^k \\
\text{subject to} & \\
& A^k x^k = b \\
& x^k \geq 0
\end{array}
\qquad\qquad
\begin{array}{ll}
\max & \pi^k b \\
\text{subject to} & \\
& \pi^k A^k \leq c^k
\end{array}
$$

Column Gen. as a Cutting Plane Method

- What is the pricing problem doing?
  - adding a constraint to the dual

$$\begin{array}{ll} \min & c^k \; x^k \\ \text{subject to} & \\ & A^k x^k = b \\ & x^k \geq 0 \end{array} \qquad \begin{array}{ll} \max & \pi^k b \\ \text{subject to} & \\ & \pi^k A^k \leq c^k \end{array}$$

Column Gen. as a Cutting Plane Method

- What is the pricing problem doing?
  - adding a constraint to the dual
- The pricing problem is a separation algorithm for the dual
  - at each iteration, it adds a new cut

$$
\begin{array}{ll}
\min & c^k \; x^k \\
\text{subject to} & \\
& A^k x^k = b \\
& x^k \geq 0
\end{array}
\qquad
\begin{array}{ll}
\max & \pi^k b \\
\text{subject to} & \\
& \pi^k A^k \leq c^k
\end{array}
$$

Column Gen. as a Cutting Plane Method

- What is the pricing problem doing?
  - adding a constraint to the dual
- The pricing problem is a separation algorithm for the dual
  - at each iteration, it adds a new cut
- The RMP may produce different dual solutions
  - what does it imply for the cuts?

Column Gen. as a Cutting Plane Method

- What is the pricing problem doing?
  - adding a constraint to the dual
- The pricing problem is a separation algorithm for the dual
  - at each iteration, it adds a new cut
- The RMP may produce different dual solutions
  - what does it imply for the cuts?
  - they may affect very different parts of the feasible space
    - especially in case of degeneracy

Stabilized Column Generation

- Goal
  - stabilizing the solution process in the dual space
    - trying to have the dual objective improve consistently
  - avoiding to jump around in the dual space
    - reducing the number of iterations

Stabilized Column Generation

- Key idea
  - ensuring that the dual values do not change too much
    - the dual improves from its current point
    - the pricing problem evolves slowly
- How?
  - enclose the dual in the box
- Two cases
  - if the solution is on the boundaries of the box
    - expand the box in that direction
  - otherwise
    - an optimal solution has been found; the box can be reduced

Perturbation Method

- Replace the primal problem by

(P_ϵ)

$$
\begin{array}{ll}
\min & c\,x \\
\text{subject to} & \\
& Ax - y^- + y+ = b \\
& y^- \leq \epsilon^- \\
& y^+ \leq \epsilon^+ \\
& x, y^-, y^+ \geq 0
\end{array}
$$

Penalty Method

- Replace the primal problem by

$$(P_\delta)$$

$$\begin{array}{ll} \min & c\ x + \delta \|Ax - b\|_1 \\ \text{subject to} & \\ & x \geq 0 \end{array}$$

Penalty Method

- This is equivalent to

$$(P_\delta)$$

$$\begin{array}{ll} \min & c\ x + \delta y^- + \delta y^+ \\ \text{subject to} & \\ & Ax - y^- + y^+ = b \\ & x, y^-, y^+ \geq 0 \end{array}$$

Penalty Method

(P_δ)

$$
\begin{array}{ll}
\text{min} & c\ x + \delta y^- + \delta y^+ \\
\text{subject to} & \\
& Ax - y^- + y^+ = b \\
& x, y^-, y^+ \geq 0
\end{array}
$$

Dual of (P_δ)

$$
\begin{array}{ll}
\text{max} & \pi b \\
\text{subject to} & \\
& \pi A \leq c \\
& -\pi \leq \delta \mathbf{1} \\
& \pi \leq \delta \mathbf{1}
\end{array}
$$

Penalty Method

(P_δ)

$$\begin{array}{ll} \min & c\,x + \delta y^- + \delta y^+ \\ \text{subject to} & \\ & Ax - y^- + y^+ = b \\ & x, y^-, y^+ \geq 0 \end{array}$$

Dual of (P_δ)

$$\begin{array}{ll} \max & \pi b \\ \text{subject to} & \\ & \pi A \leq c \\ & -\pi \leq \delta \mathbf{1} \\ & \pi \leq \delta \mathbf{1} \end{array}$$

- Observe that

$$-\delta \leq \pi_i \leq \delta$$

Stabilized Column Generation

- Key idea
  - combine the perturbation and penalty methods

Stabilized Column Generation

- Key idea
  - combine the perturbation and penalty methods

$$(\tilde{P})$$

$$
\begin{array}{ll}
\min & c\,x - \delta^- y^- + \delta^+ y^+ \\
\text{subject to} & \\
 & Ax - y^- + y^+ = b \\
 & y^- \leq \epsilon^- \\
 & y^+ \leq \epsilon^+ \\
 & x, y^-, y^+ \geq 0
\end{array}
$$

$$(\tilde{D})$$

$$
\begin{array}{ll}
\max & \pi b - \epsilon^- w^- - \epsilon^+ w^+ \\
\text{subject to} & \\
 & \pi A \leq c \\
 & \delta^- \leq \pi + w^- \\
 & \delta^+ \geq \pi - w^+ \\
 & w^-, w^+ \geq 0
\end{array}
$$

Some Intuition

$(\tilde{P})$

$$
\begin{array}{ll}
\min & c\,x - \delta^- y^- + \delta^+ y^+ \\
\text{subject to} & \\
& Ax - y^- + y^+ = b \\
& y^- \leq \epsilon^- \\
& y^+ \leq \epsilon^+ \\
& x, y^-, y^+ \geq 0
\end{array}
$$

$(\tilde{D})$

$$
\begin{array}{ll}
\max & \pi b - \epsilon^- w^- - \epsilon^+ w^+ \\
\text{subject to} & \\
& \pi A \leq c \\
& \delta^- \leq \pi + w^- \\
& \delta^+ \geq \pi - w^+ \\
& w^-, w^+ \geq 0
\end{array}
$$

- The stabilized column generation penalizes the dual variables
  - when they are outside the box $[\delta^-, \delta^+]$

Some Intuition

$$(\tilde{P})$$

$$\begin{array}{ll} \min & c\,x - \delta^- y^- + \delta^+ y^+ \\ \text{subject to} & \\ & Ax - y^- + y^+ = b \\ & y^- \leq \epsilon^- \\ & y^+ \leq \epsilon^+ \\ & x, y^-, y^+ \geq 0 \end{array}$$

$$(\tilde{D})$$

$$\begin{array}{ll} \max & \pi b - \epsilon^- w^- - \epsilon^+ w^+ \\ \text{subject to} & \\ & \pi A \leq c \\ & \delta^- \leq \pi + w^- \\ & \delta^+ \geq \pi - w^+ \\ & w^-, w^+ \geq 0 \end{array}$$

- When choosing ϵ^- and ϵ^+ small enough, this will give primal and dual solutions close to Problem P

Some Intuition

- Problems P and $\tilde{P}$ are equivalent when
  - $\epsilon^- = \epsilon^+ = 0$
  - $\delta^- < \pi < \delta^+$
- These gives stopping criteria for the algorithm
- Observe also that

$$\mathcal{O}(\tilde{P}) \leq \mathcal{O}(P)$$

Column Generation

- Restricted Master Problem

$$\begin{array}{ll} \min & c^k \; x^k \\ \text{subject to} & \\ & A^k x^k = b \\ & x^k \geq 0 \end{array} \qquad \begin{array}{ll} \max & \pi^k b \\ \text{subject to} & \\ & \pi^k A^k \leq c^k \end{array}$$

Column Generation

Algorithm 1: STABILIZED COLUMNGENERATION

k = 0
while *true* **do**
  $(x^k, \pi^k) = \text{OPTIMIZE}(P^k)$
  (a,c) = PRICE(π^k)
  if $a\pi^k \leq c$ **then**
    $x^* = x^k$
    break;
  $(A^{k+1}; c^{k+1}) = (A^k, a; c^k, c)$
  k = k + 1

Stabilized Column Generation

- Key idea
  - combine the perturbation and penalty methods

$$(\tilde{P})$$

$$\begin{array}{ll} \min & c\,x - \delta^- y^- + \delta^+ y^+ \\ \text{subject to} & \\ & Ax - y^- + y^+ = b \\ & y^- \leq \epsilon^- \\ & y^+ \leq \epsilon^+ \\ & x, y^-, y^+ \geq 0 \end{array}$$

$$(\tilde{D})$$

$$\begin{array}{ll} \max & \pi b - \epsilon^- w^- - \epsilon^+ w^+ \\ \text{subject to} & \\ & \pi A \leq c \\ & \delta^- \leq \pi + w^- \\ & \delta^+ \geq \pi - w^+ \\ & w^-, w^+ \geq 0 \end{array}$$

Stabilized Column Generation

Algorithm 1: Stabilized ColumnGeneration

k = 0
while *true* **do**
  $(x^k, \pi^k) = \text{OPTIMIZE}(P^k)$
  (a,c) = $\text{PRICE}(\pi^k)$
  if $a\pi^k \leq c$ *and* $y^- = y^+ = 0$ **then**
    $x^* = x^k$
    break;
  $(A^{k+1}; c^{k+1}) = (A^k, a; c^k, c)$
  $\delta^{k+1} = \text{UPDATE-DELTA}(\ldots)$
  $\epsilon^{k+1} = \text{UPDATE-EPSILON}(\ldots)$
  k = k + 1

everything needed

Dual Stabilization

- Objective
  - Stay close to the current dual solution
- Key idea

$$(\delta^-)^{k+1} = (\delta^+)^{k+1} = \pi^k$$

- Variant

$$(\delta^-)^{k+1} + \epsilon = (\delta^+)^{k+1} - \epsilon = \pi^k$$

Simple and Effective Strategy

- Increase ϵ^- and ϵ^+
  - whenever the pricing subproblem gives new columns
- When no column can be generated
  - Reduce ϵ^- and ϵ^+ substantially
  - use π^k as the new value of δ^- and δ^+


"That's all folks!"