15-281 Fall 2023 Lecture Activity

Learning Objectives

- To model a problem as a constraint satisfaction problem
- To practice running backtracking search, forward checking, AC-3, and MRV

Q1. Course Scheduling

Suppose we have 4 classrooms (Room A, B, C, D) to fit 4 courses' office hours (112, 122, 151, 281). Each of the classes should take place in different rooms, and each course should use exactly 1 room. Additionally, the expected number of students for the course should not exceed the capacity of the room. Finally, some of the professors have preferences about the rooms they teach in.

- 112 wants to be in rooms A or D
- 281 wants to be in rooms C or D

Rooms	Capacities
A	50
B	35
C	24
D	40

Course	OH size
112	45
122	30
151	15
281	20

(a) What are the variables? What are the values? I would do the variables as the classes and the domains as the rooms because it is clearer to me that the rooms look like a domain. The other way is also ok.

(b) What are the unary constraints in this problem? What are the binary constraints? Draw the constraint graph. Unary constraints are the room capacities/course sizes and the professor's preference constraints. The binary constraints are the all-different constraints. The graph is fully connected.

(c) When we create the problem, each variable has a domain of size 4. Enforce unary constraints to remove values that could never be assigned to each variable.

Variables	Domains after removal from unary constraints
112	{A}
122	{A, B, D}
151	{A, B, C, D}
281	{C, D}

Now, use your answers from the previous part to run backtracking search.

(d) First run backtracking search with no filtering (i.e., no Forward checking or AC-3).

Without any more information, we can choose the variables in any order. Let's work with the largest course number to smallest.

We select our first variable 281.

We choose one value to try to assign 281 first – C.

We then recurse.

> We select our second variable 151.
>
> We assign 151 to A. We then recurse.
>
> > We select our third variable 122.
> >
> > We try to assign 122 A but it doesn't pass the constraint check (A already assigned).
> >
> > We assign 122 B. We then recurse.
> >
> > > We select our fourth variable 112.
> > >
> > > We try to assign 112 A but it doesn't pass the constraint check (A already assigned).
> > >
> > > There are no other options for 112 so we return fail.
> >
> > We assign 122 D. We then recurse.
> >
> > > We select our fourth variable 112.
> > >
> > > We try to assign 112 A but it doesn't pass the constraint check (A already assigned).
> > >
> > > There are no other options for 112 so we return fail.
> >
> > There are no other options for 122 so we return fail.
>
> We assign 151 to B. We then recurse.
>
> > We select our third variable 122.
> >
> > We try to assign 122 A. We then recurse.
> >
> > > We select our fourth variable 112.
> > >
> > > We try to assign 112 A but it doesn't pass the constraint check (A already assigned).
> > >
> > > There are no other options for 112 so we return fail.
> >
> > We try to assign 122 B but it doesn't pass the constraint check (B already assigned).
> >
> > We try to assign 122 D. We then recurse.
> >
> > > We select our fourth variable 112.
> > >
> > > We try to assign 112 A and it succeeeds.

The final assignment is {112: A, 122: D, 151: B, 281: C}. We had to backtrack 4 times.

(e) Perform the backtracking search with Forward checking.

Without any more information, we can choose the variables in any order. Let's work with the largest course number to smallest.

We select our first variable 281.

We choose one value to try to assign 281 first – C.

We check all constraints that point towards 281.
112 → 281: A can still be assigned to 112 if 281 is C.
122 → 281: All values can still be assigned.
151 → 281: We must remove C from the domain of 151. New domain: {A, B, D}

We then recurse.

> We select our second variable 151.
>
> We assign 151 to A.
>
> We check all constraints that point towards 151.
>
> 112 → 151: A can NOT still be assigned to 112 if 151 is A.
>
> Remove A from 112 domain. Empty domain. FAIL; put A back in 112 domain.
>
> We assign 151 to B.
>
> We check all constraints that point towards 151.
>
> 112 → 151: A can still be assigned to 112 if 151 is B.
>
> 122 → 151: Remove B from 122's domain. New domain: {A, D}
>
> We then recurse.
>
>> We select our third variable 122.
>>
>> We try to assign 122 A.
>>
>> We check all constraints that point towards 122.
>>
>> 112 → 122: A can NOT still be assigned to 112 if 122 is A.
>>
>> Remove A from 112 domain. Empty domain. FAIL; put A back in 112 domain.
>>
>> We try to assign 122 D.
>>
>> We then recurse.
>>
>>> We select our fourth variable 112.
>>>
>>> We try to assign 112 A and it succeeeds.

The final assignment is {112: A, 122: D, 151: B, 281: C}. We never backtracked, but we spent more time checking domains.

(f) Perform the backtracking search with AC-3.

Without any more information, we can choose the variables in any order. Let's work with the largest course number to smallest.

We select our first variable 281.

We choose one value to try to assign 281 first – C.

We check all constraints that point towards 281 (the queue).
112 $\rightarrow$ 281: A can still be assigned to 112 if 281 is C.
122 $\rightarrow$ 281: All values can still be assigned.
151 $\rightarrow$ 281: We must remove C from the domain of 151. New domain: {A, B, D}
Add all edges that point towards 151 to the queue. 112 $\rightarrow$ 151: A can still be assigned if B or D are selected for 151. 122 $\rightarrow$ 151: A value can still be assigned to 122 with all values of 151. 281 $\rightarrow$ 151: Same here.

We then recurse.

> We select our second variable 151.
>
> We assign 151 to A.
>
> We check all constraints that point towards 151.
>
> 112 $\rightarrow$ 151: A can NOT still be assigned to 112 if 151 is A.
>
> Remove A from 112 domain. Empty domain. FAIL; put A back in 112 domain.
>
> We assign 151 to B.
>
> We check all constraints that point towards 151.
>
> 112 $\rightarrow$ 151: A can still be assigned to 112 if 151 is B.
>
> 122 $\rightarrow$ 151: Remove B from 122's domain. New domain: {A, D}. Add to the queue.
>
> 112 $\rightarrow$ 122: If 112 is assigned A then 151 could be assigned D.
>
> 151 $\rightarrow$ 122: If 151 is B, then 122 could be either A or D.
>
> We then recurse.
>
> > We select our third variable 122.
> >
> > We try to assign 122 A.
> >
> > We check all constraints that point towards 122.
> >
> > 112 $\rightarrow$ 122: A can NOT still be assigned to 112 if 122 is A.
> >
> > Remove A from 112 domain. Empty domain. FAIL; put A back in 112 domain.
> >
> > We try to assign 122 D.
> >
> > We then recurse.
> >
> > > We select our fourth variable 112.
> > >
> > > We try to assign 112 A and it succeeds.

The final assignment is {112: A, 122: D, 151: B, 281: C}. We never backtracked, but we spent more time checking domains.

(g) Perform the backtracking search with AC-3 and MRV (minimum remaining values).

We need to choose variables to work on based on those with the minimum remaining values in the domain.

We select our first variable 112.

We assign 112 to A (its only possible value).

We check all constraints that point towards 112 (the queue).
122 → 112: Remove A from domain of 122. Add edges with head to 122 to queue.
151 → 112: Remove A from domain of 151. Add edges with head to 151 to queue
281 → 112: C and D are both still valid.
281 → 122: C and D are both still valid.
151 → 122: B, C, and D are all still valid.
112 → 122: A can still be assigned if 122 is B or D.
112 → 151: A can still be assigned if B, C, D are selected for 151. 122 → 151: B and D can still be assigned if 151 is B, C, or D. 281 → 151: Same here.

We then recurse.

> We select our second variable. Both 122 and 281 have two variables.
>
> We assign 122 to B.
>
> We check all constraints that point towards 122.
>
> 281 → 122: C and D are both still valid.
>
> 151 → 122: Remove B from 151. New domain {C,D}. Add edges with head to 151 to queue.
>
> 112 → 122: A can still be assigned if 122 is B.
>
> 112 → 151: Still valid.
>
> 122 → 151: Still valid.
>
> 281 → 151: C and D could both be assigned.
>
> > We select our third. Two have domains of size 2. Choose variable 151 arbitrarily.
> >
> > We try to assign 151 C.
> >
> > We check all constraints that point towards 151.
> >
> > 112 → 151: Still valid.
> >
> > 122 → 151: Still valid.
> >
> > 281 → 151: C cannot be assigned to 281. Remove C from 281's domain. New domain {D}.
> >
> > Add all edges that point to 281. They are all still valid since everything is already assigned.
> >
> > We then recurse.
> >
> > > We select our fourth variable 281.
> > >
> > > We try to assign 281 D and it succeeds.

The final assignment is {112: A, 122: B, 151: C, 281: D}. We never backtracked, but we spent more time checking domains.

Using LCV to select which value to assign to a variable requires running forward checking to see how many values are removed from other domains if that value is assigned to the variable.