ECS 20: Discrete Mathematics for Computer Science

Winter 2021

Ji Wang

Week 5, February 1

Outline

- Relations in Computer Science
- Propositional Logic in Computer Science
- Proof Techniques Recap

Relations in Computer Science

Suppose A is a set of students' names, B is a set of of student IDs, can we define a relation R that specifies all the currently enrolled students at UC Davis?

Relations in Computer Science

Suppose A is a set of students' names, B is a set of of student IDs, can we define a relation R that specifies all the currently enrolled students at UC Davis?

Yes, we can. $R = \{(\text{Ji Wang}, 912345678), (\text{Jayneel Vora}, 923456789), (\text{Parichaya Chatterji}, 934567890), \cdots\}$

Relations in Computer Science

Suppose A is a set of students' names, B is a set of of student IDs, can we define a relation R that specifies all the currently enrolled students at UC Davis?

Yes, we can. $R = \{(\text{Ji Wang}, 912345678), (\text{Jayneel Vora}, 923456789), (\text{Parichaya Chatterji}, 934567890), \cdots\}$

However, more often, we want more information about a student, for instance, undergrad/graduate, department, etc. **Can we extend relation R to n-ary?**

Relations in Computer Science

Definition. Let $A_1, A_2, \cdots, A_n$ be sets. An *n*-ary relation on these sets is a subset of $A_1 \times A_2 \times \cdots \times A_n$.

Relations in Computer Science

Definition. Let $A_1, A_2, \cdots, A_n$ be sets. An n-ary relation on these sets is a subset of $A_1 \times A_2 \times \cdots \times A_n$.

App: Relational Databases [1]

1. Present the data to the user as relations;
2. Provide relational operators to manipulate the data in tabular form.

In practice, we organize data into one or more tables (or "relations") of *columns and rows*, with a unique key identifying each row. Rows are also called records or tuples while columns are also called attributes or fields.

[1] Wikipedia

Relations in Computer Science

Example. Think about class roster. If you're assigned to design the table to maintain class roster, what are necessary fields?

Relations in Computer Science

Example. Think about class roster. If you're assigned to design the table to maintain class roster, what are necessary fields?

Name	ID	Kerberos	Section	Major	$\cdots$	Grade

Relations in Computer Science

Example. Think about class roster. If you're assigned to design the table to maintain class roster, what are necessary fields?

Name	ID	Kerberos	Section	Major	$\cdots$	Grade
Rachel	932221234	rgreen	A01	MGT	$\cdots$	A-
Monica	932221235	mgeller	A01	FST	$\cdots$	A+
Phoebe	921119876	pbuffay	A03	PSC	$\cdots$	B+
Joey	920001234	jtri	A03	DRA	$\cdots$	A-
Chandler	913339876	cbing	A02	ECS	$\cdots$	A
Ross	913339877	rgeller	A02	BIS	$\cdots$	A+

Manipulate data by **SQL** (Structured Query Language), e.g. Addition, Deletion, Update and Search.

Relations in Computer Science (More in ECS-165)

Name	ID	Kerberos	Section	Major	⋯	Grade
Rachel	932221234	rgreen	A01	MGT	⋯	A-
Monica	932221235	mgeller	A01	FST	⋯	A+
Phoebe	921119876	pbuffay	A03	PSC	⋯	B+
Joey	920001234	jtri	A03	DRA	⋯	A-
Chandler	913339876	cbing	A02	ECS	⋯	A
Ross	913339877	rgeller	A02	BIS	⋯	A+

1. Add student Gunther (no longer on the waitlist):
   `INSERT INTO ecs20_roster VALUES (‘Gunther’, 931234567, ‘gcentral’, ···);`
2. Delete Rachel’s record (she drops):
   `DELETE FROM ecs20_roster WHERE Name=‘Rachel’;`
3. Update Joey’s grade to A:
   `UPDATE ecs20_roster SET Grade = ‘A’ WHERE Name =‘Joey’;`
4. Search for all the students in Section A03:
   `SELECT * FROM ecs20_roster WHERE Section = ‘A03’;`

Propositional Logic in Computer Science

1. in Database Systems (Boolean Search)
2. in Programming languages
3. in Computer Architecture (Logical Circuit)
4. $\cdots$

Propositional Logic in Computer Science

Name	ID	Kerberos	Section	Major	⋯	Grade
Rachel	932221234	rgreen	A01	MGT	⋯	A-
Monica	932221235	mgeller	A01	FST	⋯	A+
Phoebe	921119876	pbuffay	A03	PSC	⋯	B+
Joey	920001234	jtri	A03	DRA	⋯	A-
Chandler	913339876	cbing	A02	ECS	⋯	A
Ross	913339877	rgeller	A02	BIS	⋯	A+

1. Search for all the students in Section A02 **AND** their grades are better than A-:
   `SELECT * FROM ecs20_roster WHERE Section = ‘A02’ AND GRADE ≥ ‘A-’;`
2. Search for all the students in Section A02 **OR** in Section A03:
   `SELECT * FROM ecs20_roster WHERE Section = ‘A02’ OR Section = ‘A03’;`
3. Search for all the students **NOT** majoring in Computer Science:
   `SELECT * FROM ecs20_roster WHERE NOT Major = ‘ECS’;`

Propositional Logic in Computer Science (More in ECS-120/140)

In many of the if conditionals and for/while loop, we might encounter propositional logic.

```
if (( a >= b && b >= c) || (b >= a && a >= c )) {
    return c;
}
else if (( a >= b && b < c ) || ( b >= a && a < c )) {
    return b;
}
else
    return a;
```

Propositional Logic in Computer Science (More in ECS-120/140)

In many of the if conditionals and for/while loop, we might encounter propositional logic.

```
if (( a >= b && b >= c) || (b >= a && a >= c )) {
    return c;
}
else if (( a >= b && b < c ) || ( b >= a && a < c )) {
    return b;
}
else
    return a;
```

$$\downarrow$$

The Boolean satisfiability problem (abbreviated SATISFIABILITY, **SAT**) is the problem of determining if there exists an interpretation that satisfies a given Boolean formula. [2]

$$(a \geq b \wedge b \geq c) \vee (b \geq a \wedge a \geq c)$$

[2] Wikipedia

Propositional Logic in Computer Science (More in ECS-154A)

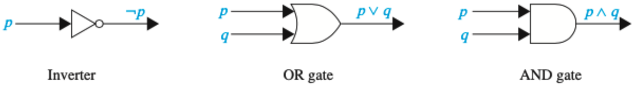

FIGURE 1 **Basic logic gates.**

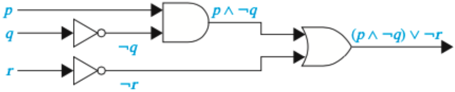

FIGURE 2 **A combinatorial circuit.**

Proof Techniques Recap

So far, we've learned:

1. Direct proof
2. Proof by contraposition
3. Proof by contradiction

New materials to come this week:

1. Equivalence proof
2. Proof by counterexample
3. **Mathematical induction**

Proof Techniques Recap

When is appropriate to use a specific proof technique?

Proof Techniques Recap

When is appropriate to use a specific proof technique?

Try direct proof first, if it's non-trivial, give contraposition or contradiction a shot!

Proof Techniques Recap

When is appropriate to use a specific proof technique?

Try direct proof first, if it's non-trivial, give contraposition or contradiction a shot!

Example 1. Prove that if m and n are integers and mn is even, then m is even or n is even.

Proof Techniques Recap

When is appropriate to use a specific proof technique?

Try direct proof first, if it's non-trivial, give contraposition or contradiction a shot!

Example 1. Prove that if m and n are integers and mn is even, then m is even or n is even.

Since there is no obvious way of showing that m is even or n is even directly from the assumption that mn is even, we attempt a proof by contraposition.

Proof. $\square$

Proof Techniques Recap

Example 2. Prove that if $5n + 4$ is odd, then n is odd.

Proof Techniques Recap

Example 2. Prove that if $5n + 4$ is odd, then n is odd.

Still, direct proof is not easy in the direction from a greater number $(5n + 4)$ to a smaller number (n).

Proof.
by Contraposition

$\square$

Proof.
by Contradiction ‘ $\square$