COMP30026
Models of Computation

Cezary Kaliszyk and William Umboh

Lecture 1

Introduction

Teaching Staff: Who Are We?

Lecturers:

- Prof. Cezary Kaliszyk
  (formal methods)
- Dr. William Umboh
  (optimization algorithms)

Head tutor:

Ari Boyd

Tutors:

Alexander Epstein, Alexander Shields, Angela Yuan, Ari Boyd, Colton Carner, Jonathan Purcell, Mark Raya, Philip Cervenjak, Rose-Maree Locsei, Samantha Tang, Tony He, Ziyu Li

Harder Algorithmic Problems

	6		1		4		5	
		8	3		5	6		
2								1
8			4		7			6
		6				3		
7			9		1			4
5								2
		7	2		6	9		
	4		5		8		7	

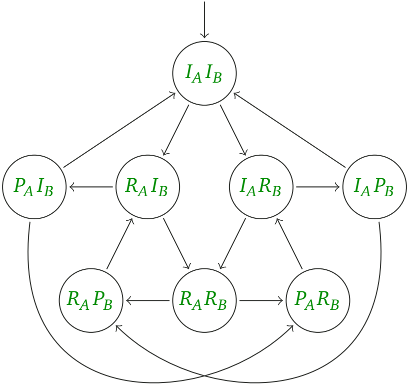
Harder Algorithmic Problems

	6		1		4		5	
		8	3		5	6		
2								1
8			4		7			6
		6				3		
7			9		1			4
5								2
		7	2		6	9		
	4		5		8		7	

- logical encoding allows to quickly develop an efficient solver for Sudoku

- similar “hard” tasks

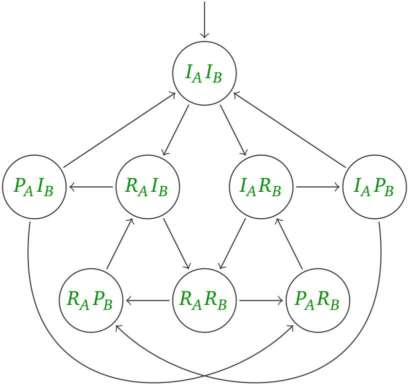
Printer Manager



two users A and B

- I_i user i is idle
- R_i print request by user i
- P_i printing document for user i

Printer Manager



two users A and B

I_i user i is idle

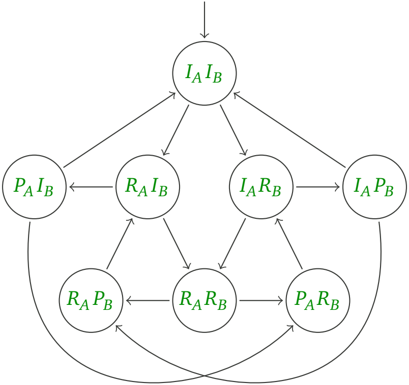
R_i print request by user i

P_i printing document for user i

some questions

- is every P_i preceded by R_i?

Printer Manager



two users A and B

I_i user i is idle

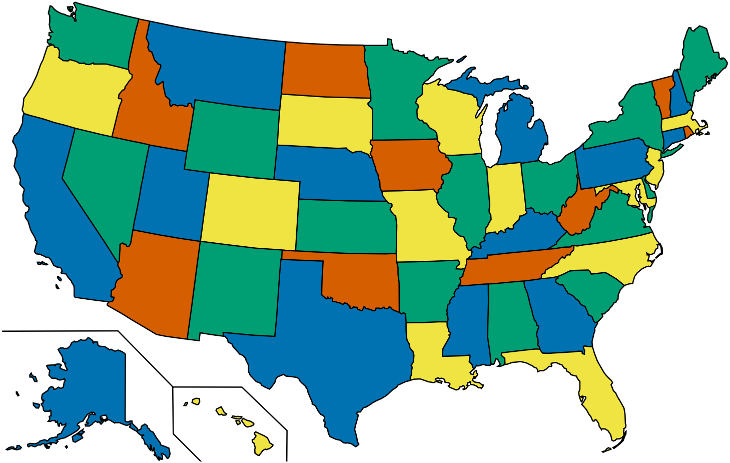
R_i print request by user i

P_i printing document for user i

some questions

- is every P_i preceded by R_i?
- is every R_i eventually followed by P_i?

Coloring a Map



Wikipedia,CC

What do we need all this “theory” for?

Before computer science, there was only mathematics.
And large parts of computer science are still very mathematical!

What do we need all this “theory” for?

Before computer science, there was only mathematics.
And large parts of computer science are still very mathematical!

Syntax: What is a well-formed program?

What do we need all this “theory” for?

Before computer science, there was only mathematics.
And large parts of computer science are still very mathematical!

Syntax: What is a well-formed program?

Type systems: What is a well-typed program?

What do we need all this “theory” for?

Before computer science, there was only mathematics.
And large parts of computer science are still very mathematical!

Syntax: What is a well-formed program?

Type systems: What is a well-typed program?

Semantics: If I run a given program, what will happen?

What do we need all this “theory” for?

Before computer science, there was only mathematics.
And large parts of computer science are still very mathematical!

Syntax:	What is a well-formed program?
Type systems:	What is a well-typed program?
Semantics:	If I run a given program, what will happen?
Specification:	What is the program supposed to do?

What do we need all this “theory” for?

Before computer science, there was only mathematics.
And large parts of computer science are still very mathematical!

Syntax: What is a well-formed program?

Type systems: What is a well-typed program?

Semantics: If I run a given program, what will happen?

Specification: What is the program supposed to do?

Verification: Prove that the program will never crash and meets its **specification**

What do we need all this “theory” for?

Before computer science, there was only mathematics.
And large parts of computer science are still very mathematical!

Syntax:	What is a well-formed program?
Type systems:	What is a well-typed program?
Semantics:	If I run a given program, what will happen?
Specification:	What is the program supposed to do?
Verification:	Prove that the program will never crash and meets its **specification**
Complexity:	How much time/memory does the program need?

What do we need all this “theory” for?

Before computer science, there was only mathematics.
And large parts of computer science are still very mathematical!

Syntax: What is a well-formed program?

Type systems: What is a well-typed program?

Semantics: If I run a given program, what will happen?

Specification: What is the program supposed to do?

Verification: Prove that the program will never crash and meets its **specification**

Complexity: How much time/memory does the program need?

And more: algorithm design, cryptography, program analysis, synthesis…

What do we need all this “theory” for?

Before computer science, there was only mathematics.
And large parts of computer science are still very mathematical!

Syntax: What is a well-formed program?

Type systems: What is a well-typed program?

Semantics: If I run a given program, what will happen?

Specification: What is the program supposed to do?

Verification: Prove that the program will never crash and meets its **specification**

Complexity: How much time/memory does the program need?

And more: algorithm design, cryptography, program analysis, synthesis…

All **used in the industry** and **active research areas**.

Topic: Automata Theory

Study of various idealized computing machines.

Automata theory subtopics:

- How they work.
- What they can do (computability theory).
- Proving elementary properties.

Topic: Formal Language Theory

Study of sets of strings.

Very close connection to automata theory.

What kinds of grammars correspond to what types of automata?

Over to You—Introductions

Please introduce yourself to your neighbours.

- where you are from?
- what degree program you are enrolled in?
- languages or programming languages that you speak?
- anything else that is interesting like: Which is the best city you have visited? Which is the greatest film ever made?

Basic mathematical vocabulary

- Natural language is bulky and often ambiguous

Basic mathematical vocabulary

- Natural language is bulky and often ambiguous
- To talk **precisely**, mathematics has its own vocabulary

Basic mathematical vocabulary

- Natural language is bulky and often ambiguous
- To talk **precisely**, mathematics has its own vocabulary
- Main items: **definitions** and **proofs**

Basic mathematical vocabulary

- Natural language is bulky and often ambiguous
- To talk **precisely**, mathematics has its own vocabulary
- Main items: **definitions** and **proofs**
- This is like learning a new language:
    - only way to learn it is to **just do it**
    - even if strange in the beginning!

Basic Symbols

Symbol	**English Reading**
$\wedge$	and (conjunction)
$\vee$	or (disjunction)
$\rightarrow$	implies (implication)
$\neg$	not (negation)

We will introduce more in the next lectures

Sudoku solution

11	12	13	14	15	16	17	18	19
21	22	23	24	25	26	27	28	29
31	32	33	34	35	36	37	38	39
41	42	43	44	45	46	47	48	49
51	52	53	54	55	56	57	58	59
61	62	63	64	65	66	67	68	69
71	72	73	74	75	76	77	78	79
81	82	83	84	85	86	87	88	89
91	92	93	94	95	96	97	98	99

	6		1		4		5	
		8	3		5	6		
2								1
8			4		7			6
		6				3		
7			9		1			4
5								2
		7	2		6	9		
	4		5		8		7	

Propositional (Boolean) Encoding

boolean x_{ijd} means field i, j has the digit d

Sudoku solution

11	12	13	14	15	16	17	18	19
21	22	23	24	25	26	27	28	29
31	32	33	34	35	36	37	38	39
41	42	43	44	45	46	47	48	49
51	52	53	54	55	56	57	58	59
61	62	63	64	65	66	67	68	69
71	72	73	74	75	76	77	78	79
81	82	83	84	85	86	87	88	89
91	92	93	94	95	96	97	98	99

	6		1		4		5	
		8	3		5	6		
2								1
8			4		7			6
		6				3		
7			9		1			4
5								2
		7	2		6	9		
	4		5		8		7	

Propositional (Boolean) Encoding

boolean x_{ijd} means field i, j has the digit d

$x_{111} \vee x_{112} \vee x_{113} \vee \ldots \vee x_{119}$

Sudoku solution

11	12	13	14	15	16	17	18	19
21	22	23	24	25	26	27	28	29
31	32	33	34	35	36	37	38	39
41	42	43	44	45	46	47	48	49
51	52	53	54	55	56	57	58	59
61	62	63	64	65	66	67	68	69
71	72	73	74	75	76	77	78	79
81	82	83	84	85	86	87	88	89
91	92	93	94	95	96	97	98	99

	6		1		4		5	
		8	3		5	6		
2								1
8			4		7			6
		6				3		
7			9		1			4
5								2
		7	2		6	9		
	4		5		8		7	

Propositional (Boolean) Encoding

boolean x_{ijd} means field i, j has the digit d

$(x_{111} \vee x_{112} \vee x_{113} \vee \ldots \vee x_{119}) \wedge (x_{121} \vee \ldots) \wedge \ldots$

Sudoku solution

11	12	13	14	15	16	17	18	19
21	22	23	24	25	26	27	28	29
31	32	33	34	35	36	37	38	39
41	42	43	44	45	46	47	48	49
51	52	53	54	55	56	57	58	59
61	62	63	64	65	66	67	68	69
71	72	73	74	75	76	77	78	79
81	82	83	84	85	86	87	88	89
91	92	93	94	95	96	97	98	99

	6		1		4		5	
		8	3		5	6		
2								1
8			4		7			6
		6				3		
7			9		1			4
5								2
		7	2		6	9		
	4		5		8		7	

Propositional (Boolean) Encoding

boolean x_{ijd} means field i, j has the digit d

$\bigwedge \{\text{at-least-one}(\{x_{ijd} \mid d \in D\}) \mid i, j \in D\}$

Sudoku solution

11	12	13	14	15	16	17	18	19
21	22	23	24	25	26	27	28	29
31	32	33	34	35	36	37	38	39
41	42	43	44	45	46	47	48	49
51	52	53	54	55	56	57	58	59
61	62	63	64	65	66	67	68	69
71	72	73	74	75	76	77	78	79
81	82	83	84	85	86	87	88	89
91	92	93	94	95	96	97	98	99

	6		1		4		5	
		8	3		5	6		
2								1
8			4		7			6
		6				3		
7			9		1			4
5								2
		7	2		6	9		
	4		5		8		7	

Propositional (Boolean) Encoding

boolean x_{ijd} means field i, j has the digit d

$\bigwedge \{\text{at-least-one}(\{x_{ijd} \mid d \in D\}) \mid i, j \in D\} \wedge$

$\bigwedge \{\text{at-most-one}(\{x_{ijd} \mid d \in D\}) \mid i, j \in D\} \wedge$

Sudoku solution

11	12	13	14	15	16	17	18	19
21	22	23	24	25	26	27	28	29
31	32	33	34	35	36	37	38	39
41	42	43	44	45	46	47	48	49
51	52	53	54	55	56	57	58	59
61	62	63	64	65	66	67	68	69
71	72	73	74	75	76	77	78	79
81	82	83	84	85	86	87	88	89
91	92	93	94	95	96	97	98	99

	6		1		4		5	
		8	3		5	6		
2								1
8			4		7			6
		6				3		
7			9		1			4
5								2
		7	2		6	9		
	4		5		8		7	

Propositional (Boolean) Encoding

boolean x_{ijd} means field i, j has the digit d

$\bigwedge \{\text{at-least-one}(\{x_{ijd} \mid d \in D\}) \mid i, j \in D\} \wedge$

$\bigwedge \{\text{at-most-one}(\{x_{ijd} \mid d \in D\}) \mid i, j \in D\} \wedge$ rows, columns, ...

Sudoku solution

11	12	13	14	15	16	17	18	19
21	22	23	24	25	26	27	28	29
31	32	33	34	35	36	37	38	39
41	42	43	44	45	46	47	48	49
51	52	53	54	55	56	57	58	59
61	62	63	64	65	66	67	68	69
71	72	73	74	75	76	77	78	79
81	82	83	84	85	86	87	88	89
91	92	93	94	95	96	97	98	99

	6		1		4		5	
		8	3		5	6		
2								1
8			4		7			6
		6				3		
7			9		1			4
5								2
		7	2		6	9		
	4		5		8		7	

Propositional (Boolean) Encoding

boolean x_{ijd} means field i, j has the digit d

$\bigwedge\{\text{at-least-one}(\{x_{ijd} \mid d \in D\}) \mid i, j \in D\} \wedge$

$\bigwedge\{\text{at-most-one}(\{x_{ijd} \mid d \in D\}) \mid i, j \in D\} \wedge$ rows, columns, ...

formula is satisfiable (can find $x_{\ldots}$) $\iff$ puzzle is solvable

Pythagorean Triples Color Problem

Can one color all natural numbers with two colors such that whenever $x^2 + y^2 = z^2$ not all of $x,\ y,\ z$ have same color?

$$3^2 + 4^2 = 5^2 \qquad 5^2 + 12^2 = 13^2 \qquad \dots$$

Pythagorean Triples Color Problem

Can one color all natural numbers with two colors such that whenever $x^2 + y^2 = z^2$ not all of x, y, z have same color?

$$3^2 + 4^2 = 5^2 \qquad 5^2 + 12^2 = 13^2 \qquad \ldots$$

Propositional (Boolean) Encoding

- propositional atoms x_i for $1 \leqslant i \leqslant n$
- $v(x_i) = \mathrm{T} \quad \Longleftrightarrow \quad$ number i is colored red
- encoding contains clauses $(x_a \vee x_b \vee x_c)$ and $(\neg x_a \vee \neg x_b \vee \neg x_c)$ for all $a^2 + b^2 = c^2$

Solution

- NO if (and only if) $n \geqslant 7825$

Solution

- NO if (and only if) $n \geqslant 7825$
- 2 days (in May 2016) on University of Texas' Stampede supercomputer with 800 processors

Solution

- NO if (and only if) $n \geqslant 7825$
- 2 days (in May 2016) on University of Texas' Stampede supercomputer with 800 processors
- 200 terabyte proof of unsatisfiability

Solution

- NO if (and only if) $n \geqslant 7825$
- 2 days (in May 2016) on University of Texas' Stampede supercomputer with 800 processors
- 200 terabyte proof of unsatisfiability
- extensive media coverage (Nature)

Learning Support

Teaching materials will be posted on the LMS every week.

Lecturers and tutors will answer your questions

- during and after class

Staff will also answer questions on (Ed) (Recommended!)

I will be running consultations every Friday, 10-11am.

Expectations

You won’t truly master problem-solving just by watching lectures.

Deliberate practice is the right way.

Attend the tutorials! They start in week 1!

Assessment

Components

- A 3-hour end-of-semester exam (70% of the final mark)
- Two assignments (due ≈ Weeks 6 and 11) (12% each)
- Weekly worksheets (12 sheets, best 9 of 12)
  - No extension to the worksheet deadlines
  - No supplementary worksheets
    That's why we have the best 9 of 12 rule.

Academic misconduct is taken seriously

- We do our best job to prevent this from happening
- Will report any potential misconduct that we find to the University for action