CSE 544
Theory of Query Languages

Announcements

- Project Milestone: was due on Friday
  - Expect feedback by the end of this week
- Project presentations: Friday, 6/7
  - Reserve all day, stay tuned for announcements
- Homework 3: was due yesterday
  - Did you remember to turn off your servers!?!
- Homework 4: will be posted in 2-3 days
- Next paper review: next Wednesday, 5/29

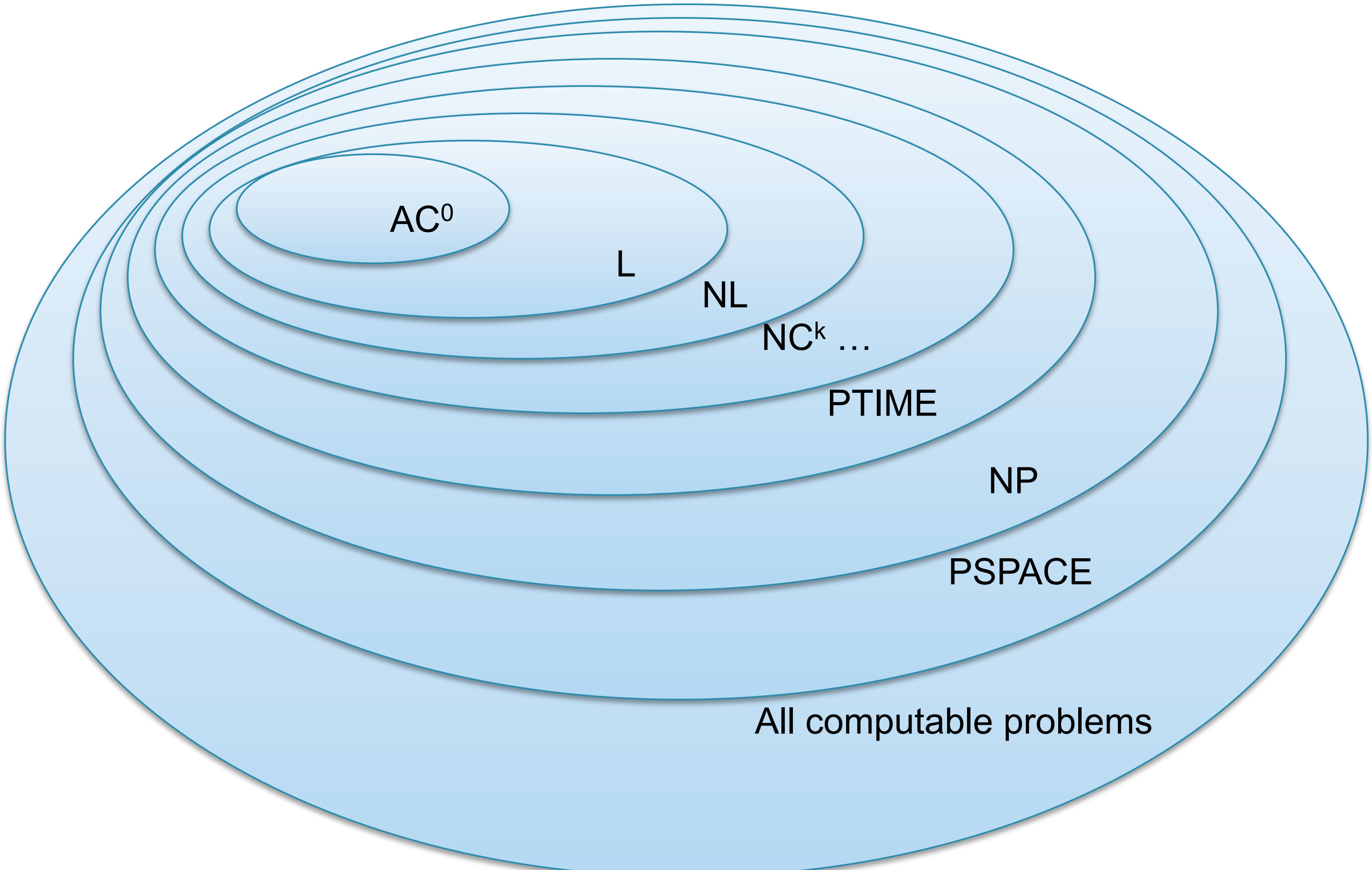
Complexity Classes

A decision problem:

- We have a property (a.k.a. problem)
- Given an input **X** of size n, decide if it satisfies the property

- In other words, define have to compute a function f(**X**) = 0 or 1

We are interested in these classes



The Class AC^0

What is AC^0 ?

The Class AC^0

A problem f is in AC^0 if, for every n, there exists a Boolean circuit s.t.:

- It consists of unbounded fan-in AND, unbounded fan-in OR, and NOT gates
- If the inputs $X_1, \ldots, X_n$ encode an input **X**, then the circuit's output is $f(\mathbf{X})$
- The circuit size is $n^{O(1)}$
- The circuit depth is $O(1)$

Example in AC^0

Problem: given an input string $X_1, \ldots, X_n$ in $\{0,1\}^n$, check if it has at least two 1's

0100101 - yes

00001000 - no

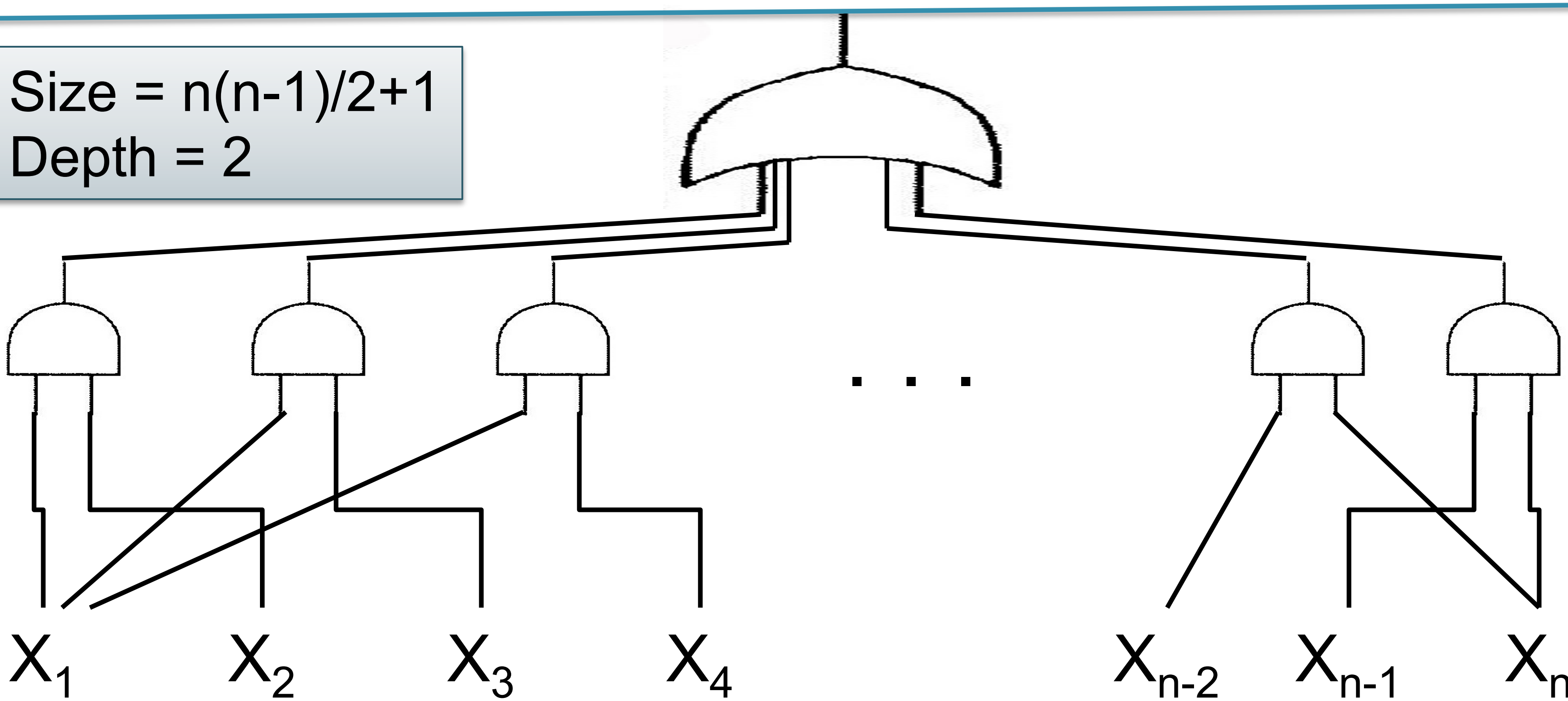
Example in AC^0

Problem: given an input string $X_1, \ldots, X_n$ in $\{0,1\}^n$, check if it has at least two 1's

0100101 - yes

00001000 - no

Size = n(n-1)/2+1
Depth = 2



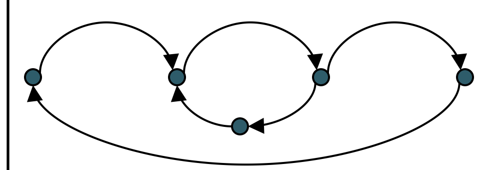
Are these in AC^0 or not?

Carry bit: The sum $(X_nX_{n-1}\ldots X_1) + (Y_nY_{n-1}\ldots Y_1)$ has a carry bit

(1001)+(0101) = (1110) NO
(1001)+(0111) = (10000) YES

Triangle: A graph given by the n × n adjacency matrix contains a directed triangle

0	1	0	0	0
0	0	1	0	0
0	0	0	1	1
1	0	0	0	0
0	1	0	0	0

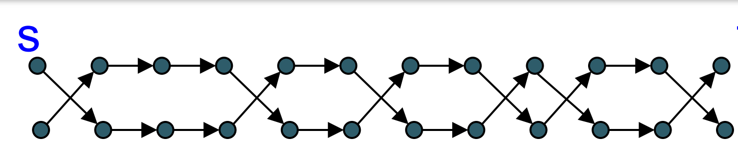


YES

Parity: $X_1,\ldots,X_n$ has an even number of 1's

10010101101 YES
10010101100 NO

s-t Reachability (GAP): A graph given by the n × n adjacency matrix contains a path from node s to node t



Are these in AC^0 or not?

Carry bit: The sum $(X_n X_{n-1} \ldots X_1) + (Y_n Y_{n-1} \ldots Y_1)$ has a carry bit

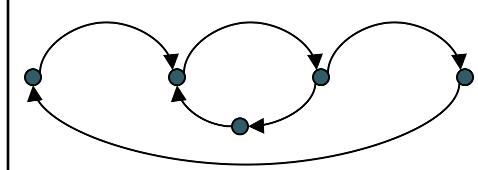
In AC^0

(1001)+(0101) = (1110) NO
(1001)+(0111) = (10000) YES

Triangle: A graph given by the n × n adjacency matrix contains a directed triangle

In AC^0

$$\begin{vmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{vmatrix}$$



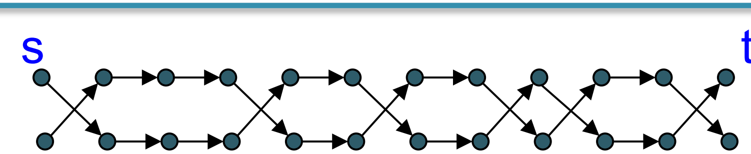
YES

Parity: $X_1,\ldots,X_n$ has an even number of 1's

Not in AC^0

10010101101 YES
10010101100 NO

s-t Reachability (GAP): A graph given by the n × n adjacency matrix contains a path from node s to node t

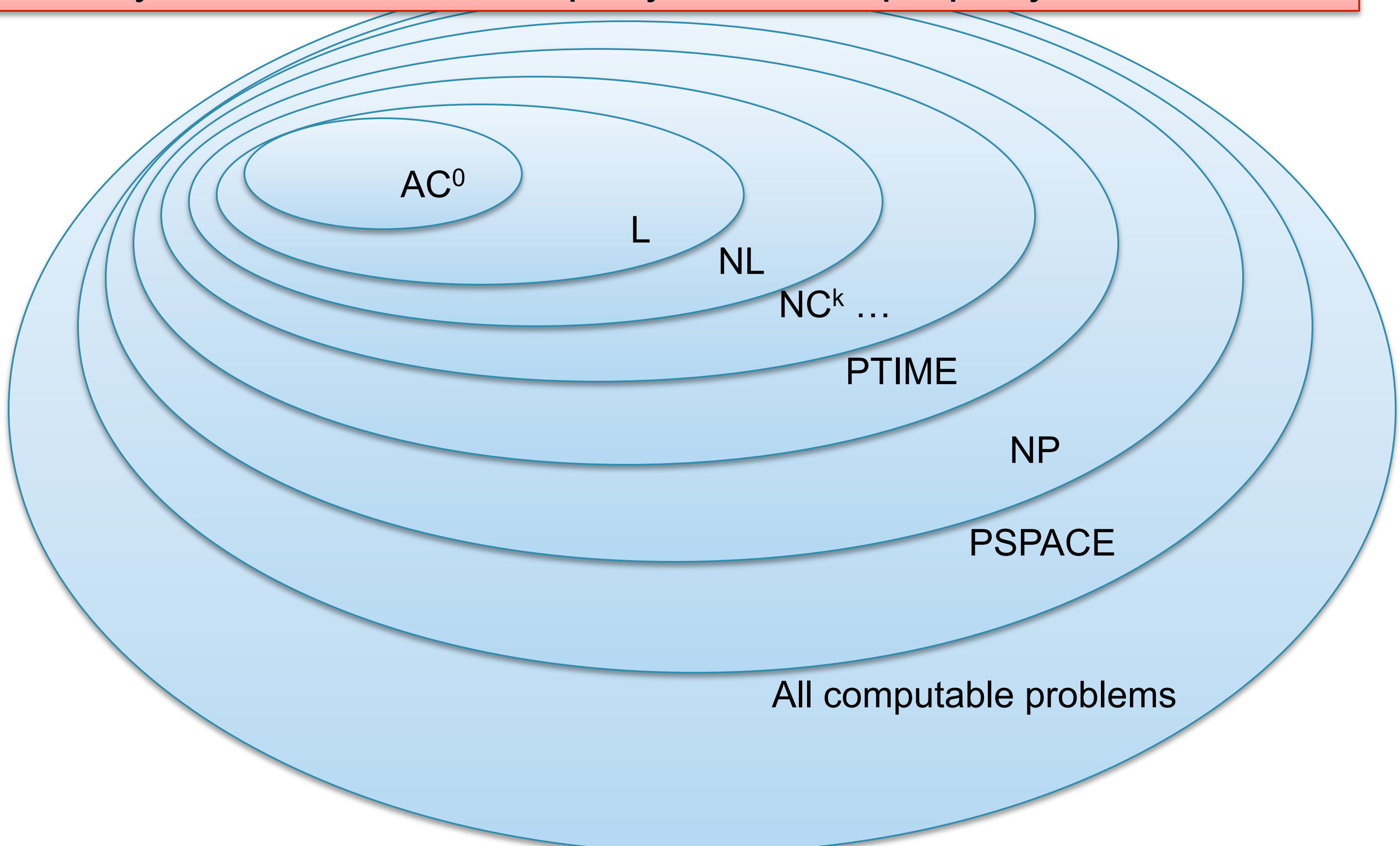


Not in AC^0

Make sure you understand why!

Relational Queries

Theorem: Every Boolean relational query defines a property that is in AC^0



```
select distinct R.A, S.C
from R, S
where R.B=S.B
```

Example

R

A	B
a	a
b	b
a	b

S

B	C
c	c
b	c

R:	a	b	c
a	1	1	0
b	0	1	0
c	0	0	0

S:	a	b	c
a	0	0	0
b	0	0	1
c	0	0	1

```
select distinct R.A, S.C
from R, S
where R.B=S.B
```

Example

R

A	B
a	a
b	b
a	b

S

B	C
c	c
b	c

$$Q = \exists z.R(\text{'a'},z) \wedge S(z,\text{'c'})$$

Prove that Q is in AC^0

Example circuit for n = 3 (i.e. ADom={a,b,c})

R:	a	b	c
a	1	1	0
b	0	1	0
c	0	0	0

S:	a	b	c
a	0	0	0
b	0	0	1
c	0	0	1

```
select distinct R.A, S.C
from R, S
where R.B=S.B
```

Example

R

A	B
a	a
b	b
a	b

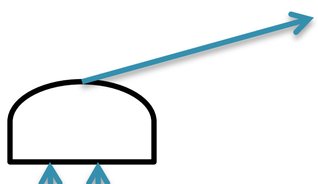
S

B	C
c	c
b	c

$$Q \;=\; \exists z.R(\text{'a'},z) \wedge S(z,\text{'c'})$$

Prove that Q is in AC^0

Example circuit for n = 3 (i.e. ADom={a,b,c})



R:

	a	b	c
a	1	1	0
b	0	1	0
c	0	0	0

S:

	a	b	c
a	0	0	0
b	0	0	1
c	0	0	1

```
select distinct R.A, S.C
from R, S
where R.B=S.B
```

Example

R

A	B
a	a
b	b
a	b

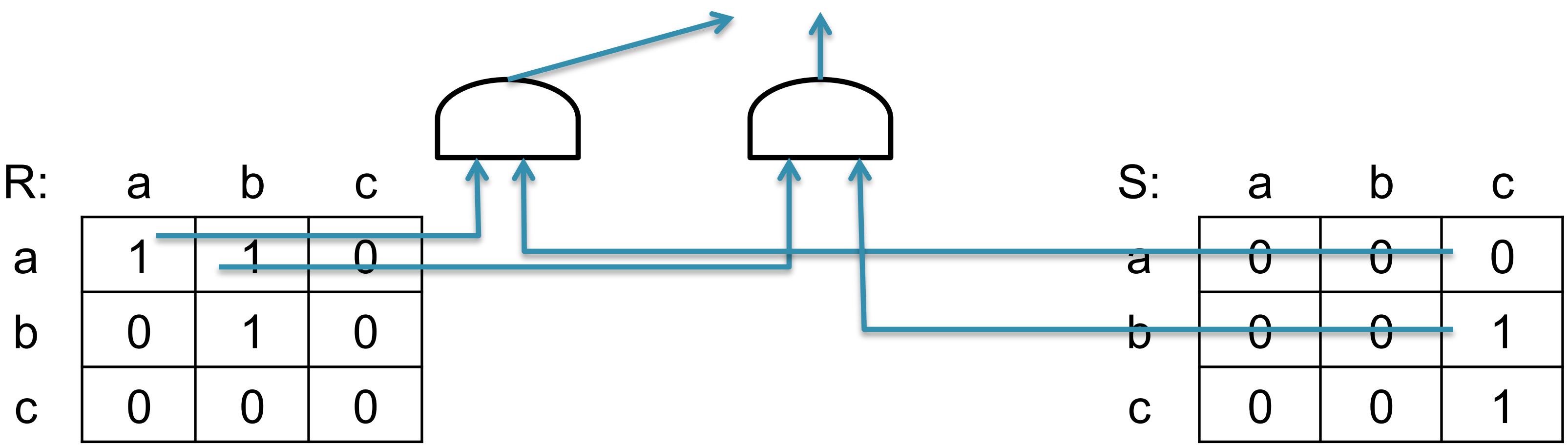
S

B	C
c	c
b	c

$Q = \exists z.R(\text{'a'},z) \wedge S(z,\text{'c'})$

Prove that Q is in AC^0

Example circuit for n = 3 (i.e. ADom={a,b,c})



R:

	a	b	c
a	1	1	0
b	0	1	0
c	0	0	0

S:

	a	b	c
a	0	0	0
b	0	0	1
c	0	0	1

Example

```
select distinct R.A, S.C
from R, S
where R.B=S.B
```

R

A	B
a	a
b	b
a	b

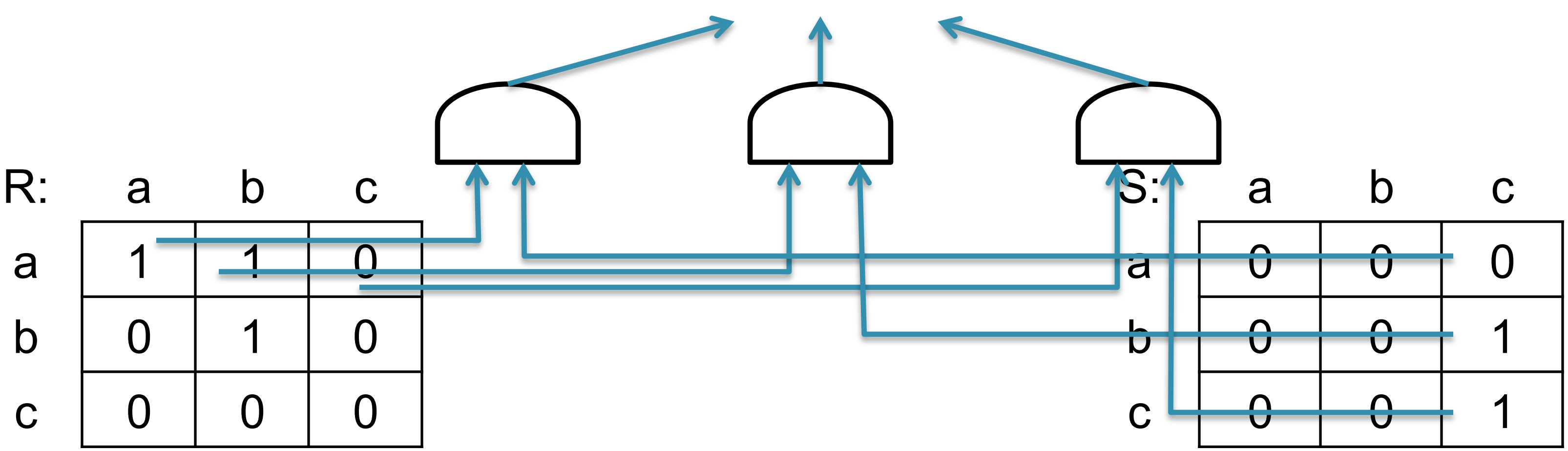
S

B	C
c	c
b	c

$$Q = \exists z.R(\text{'a'},z) \wedge S(z,\text{'c'})$$

Prove that Q is in AC^0

Example circuit for n = 3 (i.e. ADom={a,b,c})



R:

	a	b	c
a	1	1	0
b	0	1	0
c	0	0	0

S:

	a	b	c
a	0	0	0
b	0	0	1
c	0	0	1

```sql
select distinct R.A, S.C
from R, S
where R.B=S.B
```

Example

R

A	B
a	a
b	b
a	b

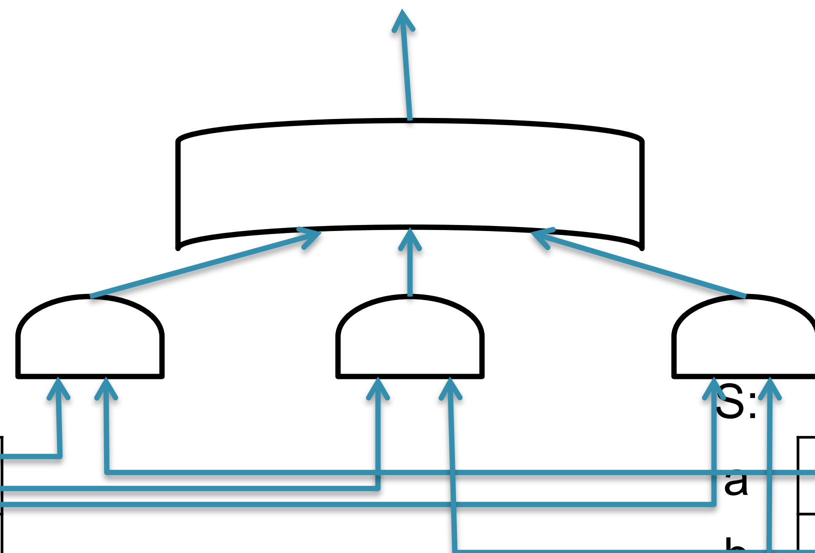
S

B	C
c	c
b	c

$$Q = \exists z.R(\text{'a'},z) \wedge S(z,\text{'c'})$$

Prove that Q is in AC^0

Example circuit for n = 3 (i.e. ADom={a,b,c})



R:

	a	b	c
a	1	1	0
b	0	1	0
c	0	0	0

S:

	a	b	c
a	0	0	0
b	0	0	1
c	0	0	1

select distinct R.A, S.C
from R, S
where R.B=S.B

Example

R

A	B
a	a
b	b
a	b

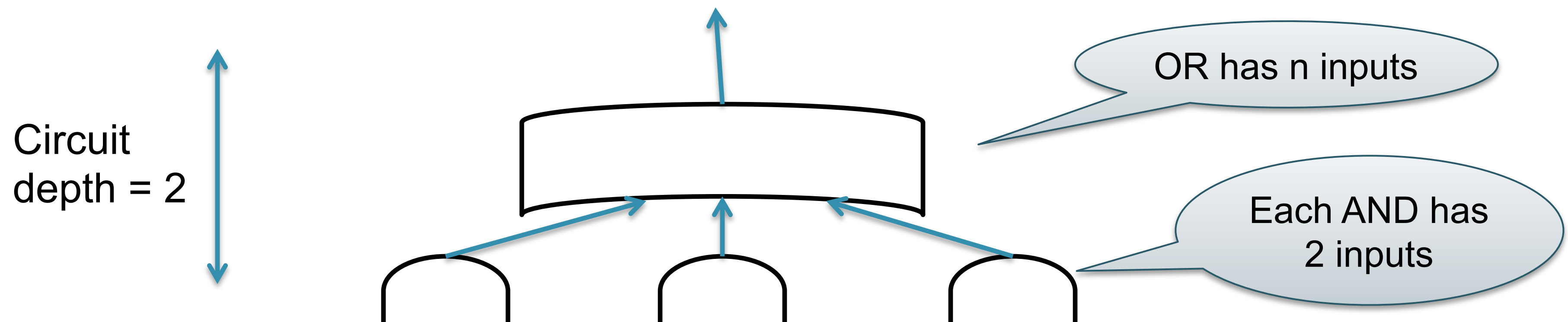
S

B	C
c	c
b	c

$$Q = \exists z. R(\text{'a'}, z) \land S(z, \text{'c'})$$

Prove that Q is in AC^0

Example circuit for n = 3 (i.e. ADom={a,b,c})



R:

	a	b	c
a	1	1	0
b	0	1	0
c	0	0	0

S:

	a	b	c
a	0	0	0
b	0	0	1
c	0	0	1

Another Example

$Q = \exists y.R(\text{'a'},y) \wedge (\forall z.S(y,z) \rightarrow \exists u.R(z,u))$

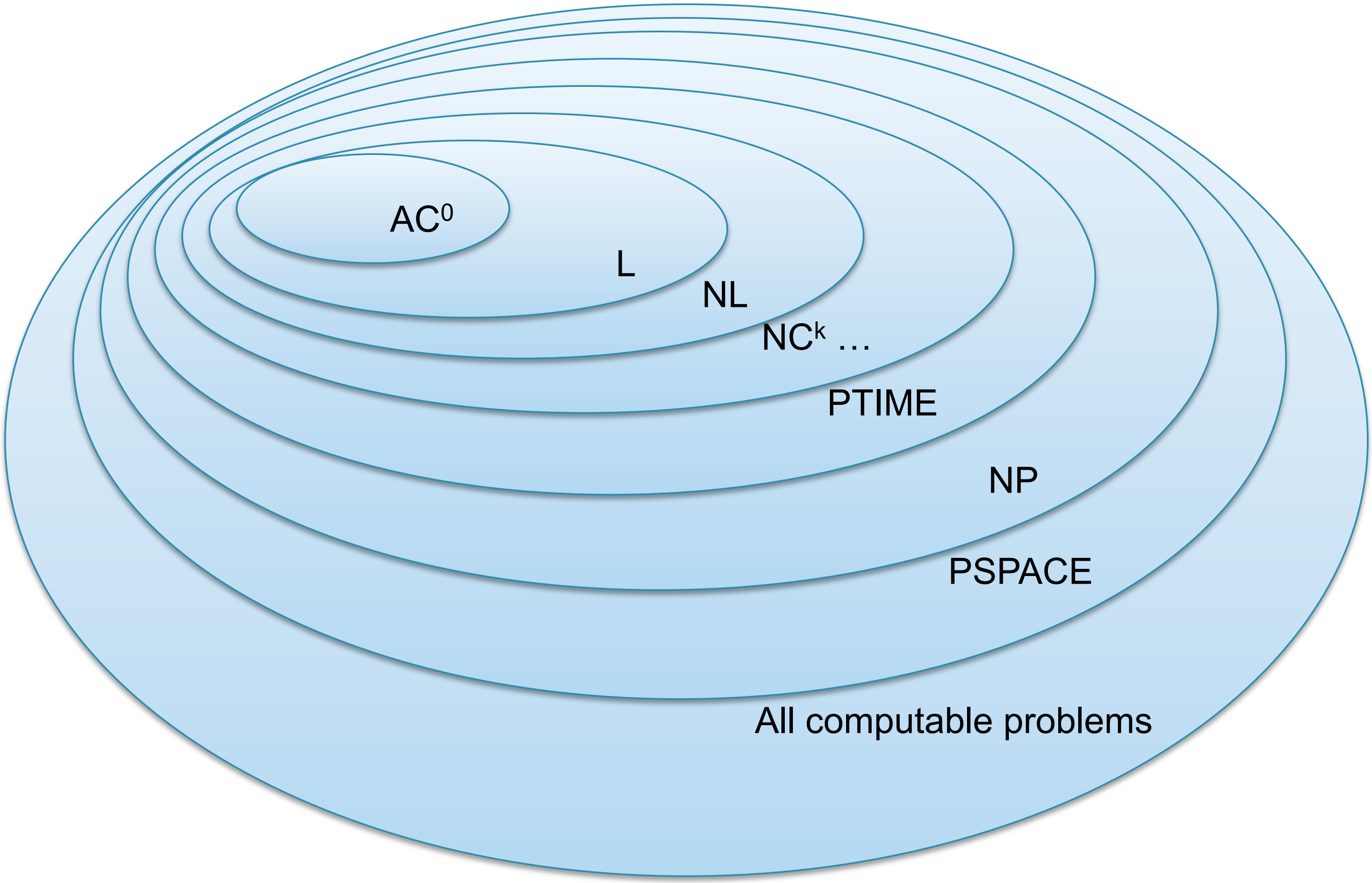
Practice at home:
Show that Q is in AC^0 by showing how to construct a circuit for computing Q.
What is the depth?
What fanouts have your OR and AND gates ?

Discussion

In class: make sure you understand very well why every relational query is in AC^0

- Consequence 1 (for theoreticians and their friends):
  - SELECT-DISTINCT-FROM-WHERE queries cannot express PARITY, GAP
- Consequence 2 (for fans of Big Data)
  - "SQL is embarrassingly parallel"

The Classes L and NL



LOGSPACE and NLOGSPACE

- What is LOGSPACE (or L) ?

- What is NLOGSPACE (or NL) ?

LOGSPACE and NLOGSPACE

- A problem is in LOGSPACE (or L) if it can be computed by a deterministic Turing machine using O(log n) space

- A problem is in NLOGSPACE (or NL) if it can be computed by a non-deterministic Turing machine using O(log n) space

O(log n) space refers to the working tape: the input is on a separate tape of size n

Examples

- GAP is in NLOGSPACE (why?)
- 1-GAP (each node has outdegree ≤ 1) is in LOGSPACE (why?)
- Recall that none of these problems is in AC^0
- Theorem: GAP is complete for NLOGSPACE
- Theorem: 1-GAP is complete for LOGSPACE

Datalog

It can express GAP

In many ways!!

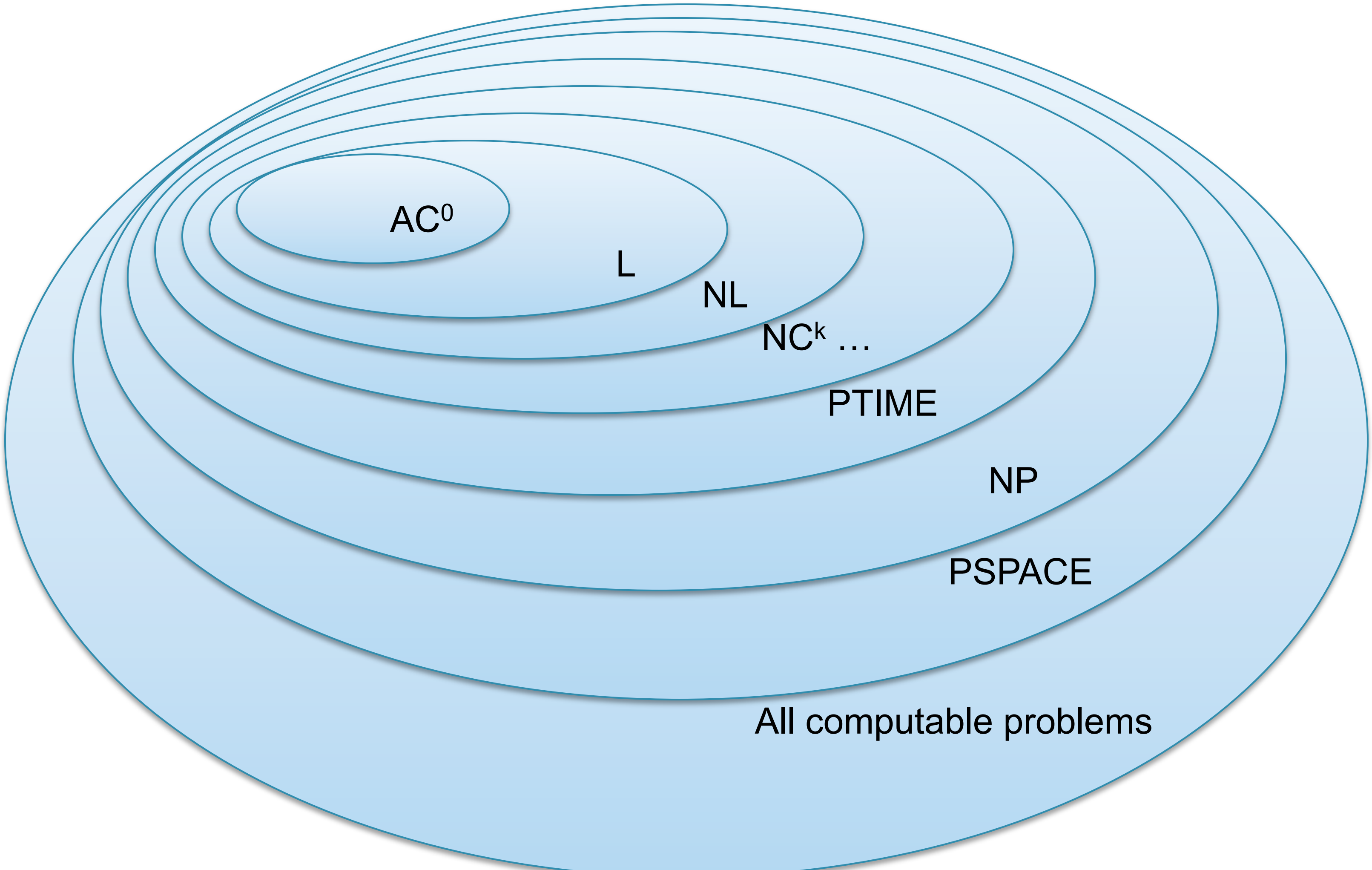
```
T(x,y) :- R(x,y)
T(x,y) :- R(x,z), T(z,y)
Answer :- T('s','t')
```

```
T(x,y) :- R(x,y)
T(x,y) :- T(x,z), R(z,y)
Answer :- T('s','t')
```

This proves that datalog can express strictly more queries than the relational calculus

```
T(x,y) :- R(x,y)
T(x,y) :- T(x,z), T(z,y)
Answer :- T('s','t')
```

The Classes PTIME, NP, PSPACE



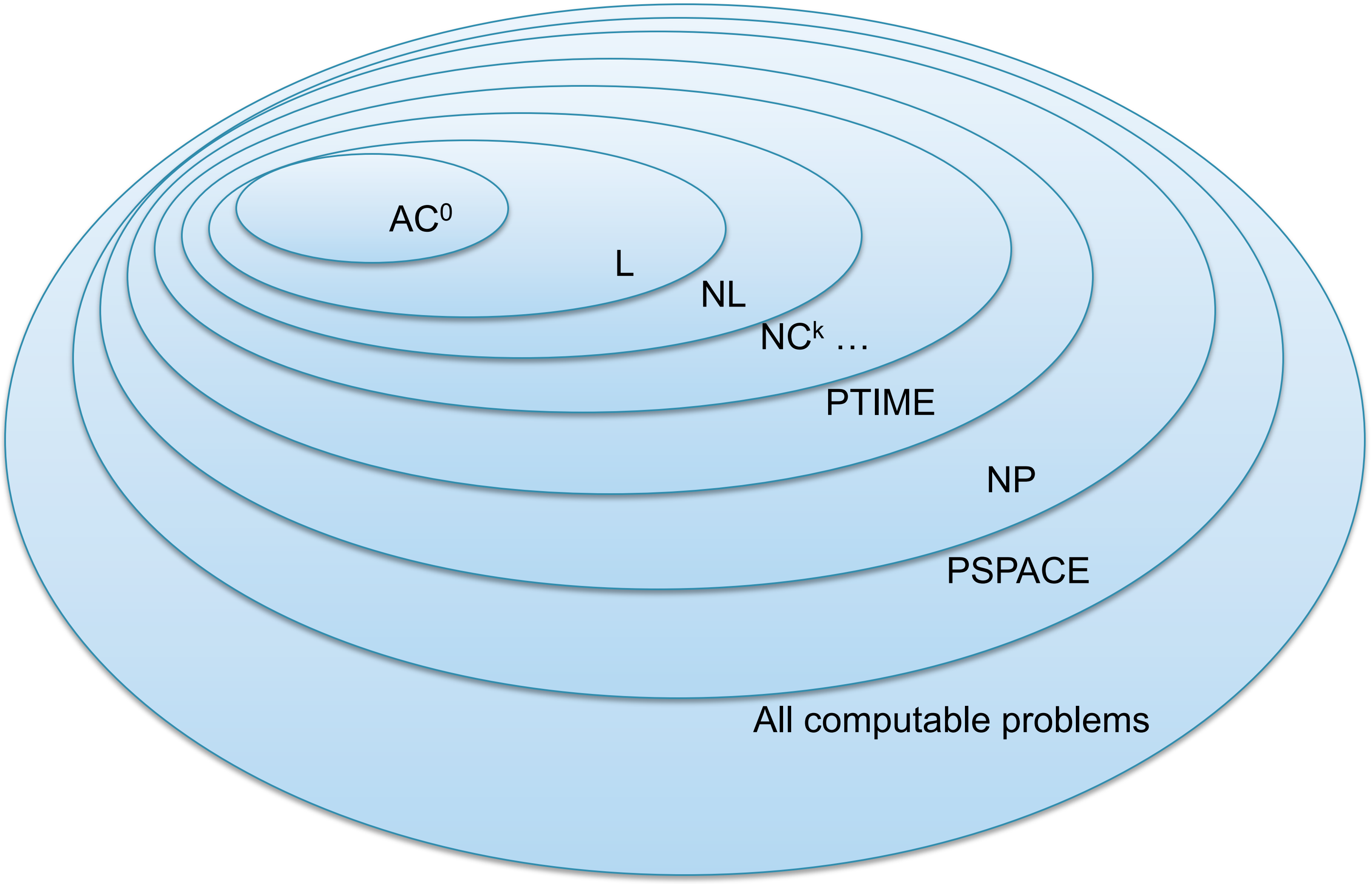
The Classes PTIME, NP, PSPACE

- What is PTIME?
- What is NP?

In class

- What is PSPACE?

What is the Complexity of Datalog?



Datalog

The Same-Generation Problem

Problem: We have a database of microbes, where each microbe x may have several children y:

Parent(x,y)

Find all microbes in the same generation with “M62251”

Can we solve it in datalog?

Is this problem in NLOGSPACE?

Datalog

The Same-Generation Problem

Problem: We have a database of microbes, where each microbe x may have several children y:

Parent(x,y)

Find all microbes in the same generation with "M62251"

Can we solve it in datalog?

Is this problem in NLOGSPACE?

YES:

```
SG(y,z) :- Parent(x,y), Parent(x,z)
SG(y,z) :- SG(u,v), Parent(u,y), Parent(v,y)
Answer(z) :- SG("M62251",z)
```

Datalog

The Same-Generation Problem

Problem: We have a database of microbes, where each microbe x may have several children y:

Parent(x,y)

Find all microbes in the same generation with "M62251"

Can we solve it in datalog?

Is this problem in NLOGSPACE?

YES:

SG(y,z) :- Parent(x,y), Parent(x,z)
SG(y,z) :- SG(u,v), Parent(u,y), Parent(v,y)
Answer(z) :- SG("M62251",z)

YES!
(in class…)

Discussion

- The same-generation problem was a trap:
  - SG is no more complex than GAP!
  - Lesson: GAP is more than meets the eyes

- But datalog *is* more expressive than NLOGSPACE: it captures all of PTIME, in ways we discuss next, in three steps

Step 1: Complexity of Datalog

<u>Theorem</u>. Datalog is in PTIME.

More precisely, fix any Boolean datalog program P.
The problem: given D, check if P(D) = true is in PTIME

Proof: … [discuss in class]

Step 1: Complexity of Datalog

Theorem. Datalog is in PTIME.

More precisely, fix any Boolean datalog program P.
The problem: given D, check if P(D) = true is in PTIME

Proof: … [discuss in class]

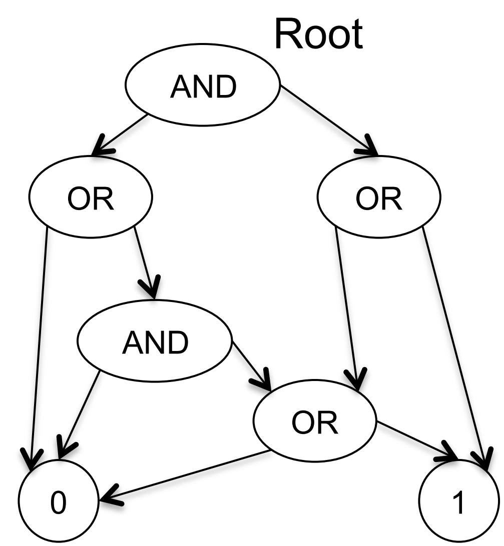
Which of the following are in PTIME?
Stratified, inflationary-fixpoint, partial-fixpoint datalog$^\neg$.

Step 2: The Circuit Value Problem

Circuit Value Problem

Input = a rooted DAG; leaves labeled 0/1, internal nodes labeled AND/OR
Output = check if the value of the root is 1

Note: NOT nodes could be added w.l.o.g. (why?)

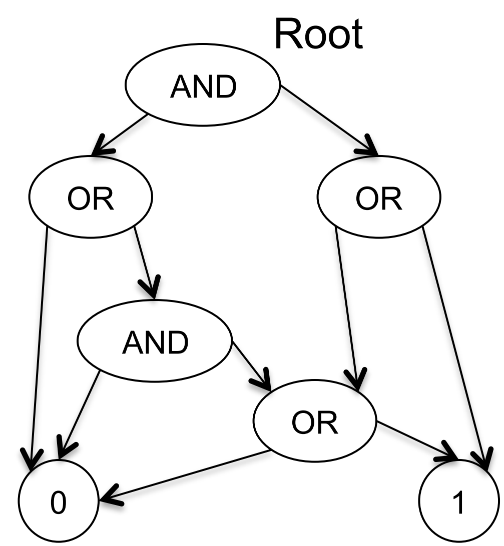


Step 2: The Circuit Value Problem

Circuit Value Problem

Input = a rooted DAG; leaves labeled 0/1, internal nodes labeled AND/OR
Output = check if the value of the root is 1

Note: NOT nodes could be added w.l.o.g. (why?)



Theorem.
The Circuit Value Problem is complete for PTIME

In class:

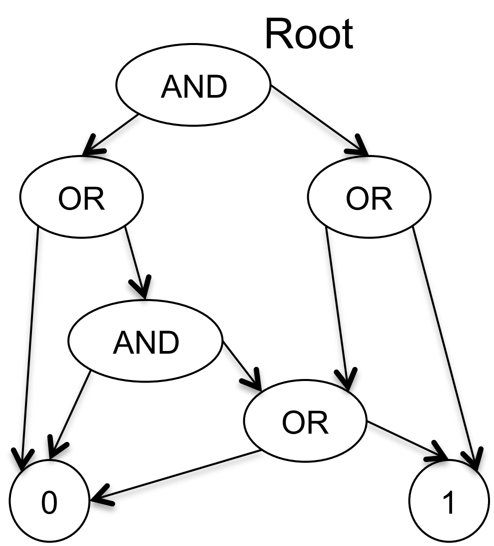
1. How can we compute it in PTIME?
2. Why isn't it in NLOGSPACE?

Step 2: The Circuit Value Problem

Theorem. Datalog can express the Circuit Value Problem

EDBs:

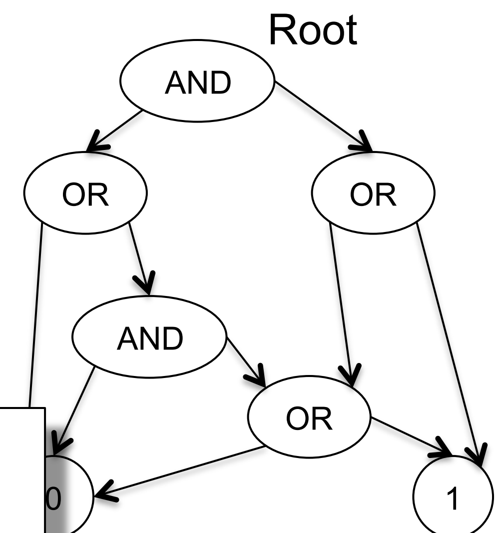
- root(x)
- and(x,y1,y2)
- or(x,y1,y2)
- zeroLeaf(x)
- oneLeaf(x)



Step 2: The Circuit Value Problem

Theorem. Datalog can express the Circuit Value Problem

EDBs:
- root(x)
- and(x,y1,y2)
- or(x,y1,y2)
- zeroLeaf(x)
- oneLeaf(x)




```
oneNode(x) :- oneLeaf(x)
oneNode(x) :- or(x,y1,y2), oneNode(y1)
oneNode(x) :- or(x,y1,y2), oneNode(y2)
oneNode(x) :- and(x,y1,y2), oneNode(y1), oneNode(y2)
Answer() :- root(x), oneNode(x)
```

Discussion

- Step 1: datalog is in PTIME
  - Stratified, and inflationary datalog$^{\neg}$ are in PTIME

- Step 2: datalog can express a PTIME complete problem

- Step 3: can datalog express *<u>all</u>* PTIME problems?

Step 3

Theorem. For every problem in PTIME there exists a program in inflationary-fixpoint datalog$^\neg$ that expresses that problem

Caveat: the program must have access to a total order on the active domain. Otherwise inflationary-fixpoint datalog$^\neg$ cannot even express parity!

Make sure you understand why pure datalog (without negation) cannot express all of PTIME

Finally: partial-fixpoint Datalog$^\neg$

Theorem. Partial-fixpoint datalog$^\neg$ can express precisely the problems that are in PSPACE

Same caveat: for completeness we need access to an order relation

Which are "Easy" to Parallelize?

- Relational calculus = AC^0
- Add transitive closure = NLOGSPACE
- Inflationary datalog = PTIME

Which are “Easy” to Parallelize?

- Relational calculus = AC^0
- YES!! “embarrassingly parallel”

- Add transitive closure = NLOGSPACE
- MAYBE: the path-doubling program

- Inflationary datalog = PTIME
- NO: circuit value problem

Descriptive Complexity

- In computational complexity one describes complexity classes in terms of a computational model
  - Turing Machine, circuit, etc
- In descriptive complexity one describes complexity classes in terms of the logic ("query language") that captures that class

Descriptive Complexity

[Immerman, Vardi]
Assume we have access to an order relation <
(and to a BIT relation for AC^0)