CSE 544: Principles of Database Systems

Query Complexity

Announcements

- Project milestone due tomorrow night
- HW3 due on Monday

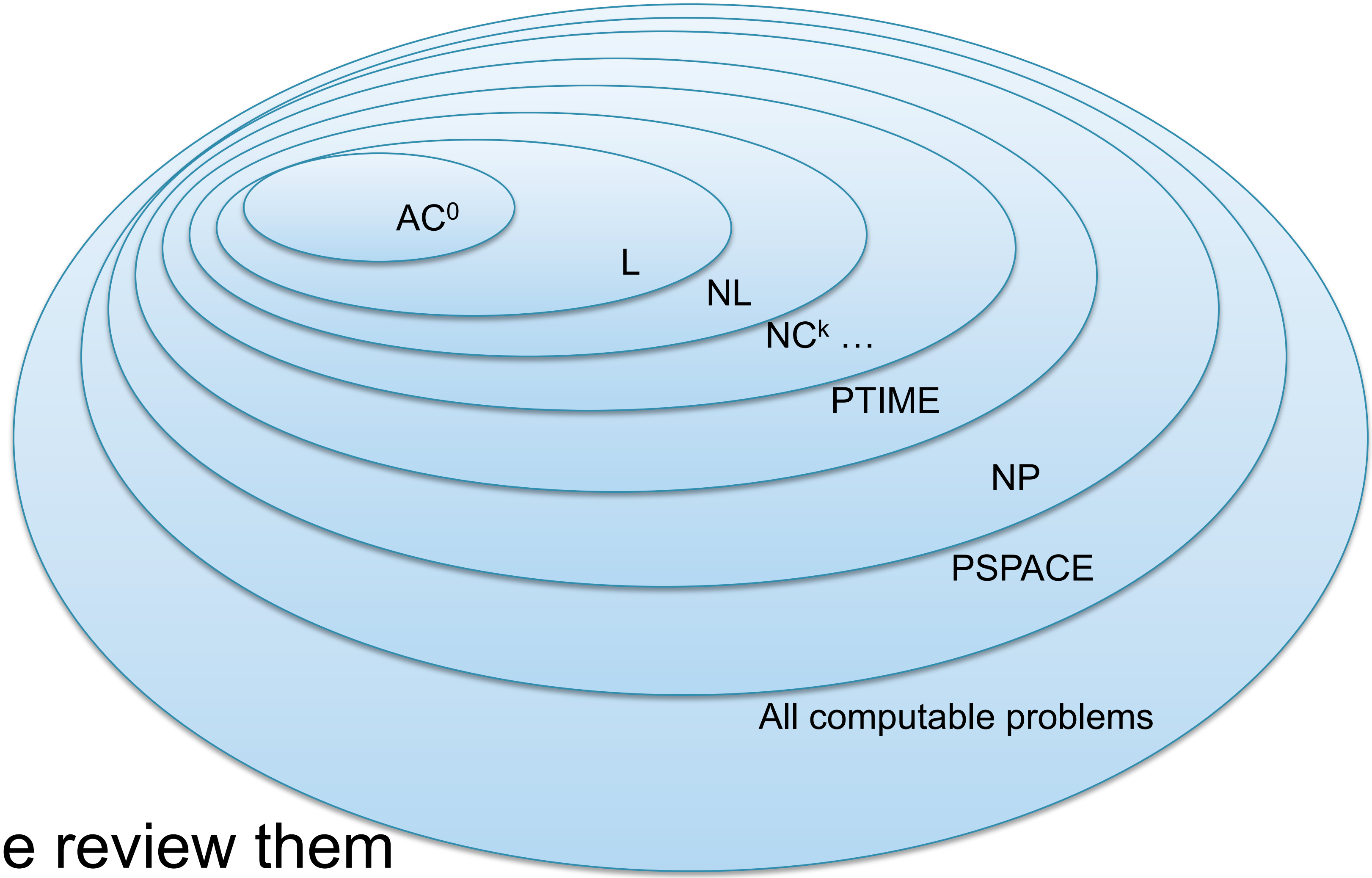
Query Complexity

Fix some query language L

- What is the complexity of evaluating queries in L on input databases?

- The answer gives us several insights:
  - What physical operators we need for L
  - What queries can and cannot we write in L
  - Whether we need to extend L

We are interested in these classes



Please review them
for next Tuesday; we will briefly discuss them in class.

Measuring Size

- The size of a database D is the number of tuples

R =

a	b
a	c
b	f
b	g
a	f

S =

c	b
a	c
a	a
b	a
b	b
g	b
f	a

- Alternatively, the size of D is the number of constants in its active domain

Number of tuples: 5+7 = 12

ADom(D)={a,b,c,d,e,f,g}
Size of the active domain = 7

These two are related (how?)

Example

$$Q(x) = \exists y.R(x,y) \wedge (\forall z.S(y,z) \rightarrow \exists u.R(z,u))$$

Database **D**:

R =

a	b
a	c
b	f
b	g
a	f

S =

c	b
a	c
a	a
b	a
b	b
g	b
f	a

Give an algorithm for computing Q on any input **D**.
Express its complexity as a function of* n = |ADom(D)|

* Active Domain = all constants in D. ADom(D) = {a,b,c,…}

Example

$$Q(x) = \exists y.R(x,y) \wedge (\forall z.S(y,z) \rightarrow \exists u.R(z,u))$$

```
for x in ADom do
   good_y = false
   for y in ADom do
      if (x,y)∈R then
         good_z = true
         for z in ADom do
            if (y,z) ∈ S then
               good_u = false
               for u in Adom do
                  if (z,u) ∈ R then good_u = true
               endfor
               if not good_u then good_z = false
         endfor
         if good_z then good_y = true
   endfor
   if good_y then output x
endfor
```

What is the running time f(n)?

In what complexity class is this query?

Example

$$Q(x) = \exists y.R(x,y) \wedge (\forall z.S(y,z) \rightarrow \exists u.R(z,u))$$

```
for x in ADom do
   good_y = false
   for y in ADom do
      if (x,y)∈R then
         good_z = true
         for z in ADom do
            if (y,z) ∈ S then
               good_u = false
               for u in Adom do
                  if (z,u) ∈ R then good_u = true
               endfor
               if not good_u then good_z = false
         endfor
         if good_z then good_y = true
   endfor
   if good_y then output x
endfor
```

What is the running time f(n)?

Complexity = $O(n^4)$

4=number of variables

In what complexity class is this query?

Example

$$Q(x) = \exists y.R(x,y) \wedge (\forall z.S(y,z) \rightarrow \exists u.R(z,u))$$

```
for x in ADom do
   good_y = false
   for y in ADom do
      if (x,y)∈R then
         good_z = true
         for z in ADom do
            if (y,z) ∈ S then
               good_u = false
               for u in Adom do
                  if (z,u) ∈ R then good_u = true
               endfor
               if not good_u then good_z = false
         endfor
         if good_z then good_y = true
   endfor
   if good_y then output x
endfor
```

What is the running time f(n)?

Complexity = $O(n^4)$

4=number of variables

In what complexity class is this query?

PTIME

Example

$$Q(x) = \exists y.R(x,y) \wedge (\forall z.S(y,z) \rightarrow \exists u.R(z,u))$$

```
for x in ADom do
   good_y = false
   for y in ADom do
      if (x,y)∈R then
         good_z = true
         for z in ADom do
            if (y,z) ∈ S then
               good_u = false
               for u in Adom do
                  if (z,u) ∈ R then good_u = true
               endfor
               if not good_u then good_z = false
         endfor
         if good_z then good_y = true
   endfor
   if good_y then output x
endfor
```

What is the running time f(n)?

Complexity = $O(n^4)$

4=number of variables

In what complexity class is this query?

F

LOGSPACE (why?)

Example

$$Q(x) = \exists y.R(x,y) \land (\forall z.S(y,z) \rightarrow \exists u.R(z,u))$$

```
for x in ADom do
   good_y = false
   for y in ADom do
      if (x,y)∈R then
         good_z = true
         for z in ADom do
            if (y,z) ∈ S then
               good_u = false
               for u in Adom do
                  if (z,u) ∈ R then good_u = true
               endfor
               if not good_u then good_z = false
         endfor
         if good_z then good_y = true
   endfor
   if good_y then output x
endfor
```

What is the running time f(n)?

Complexity = $O(n^4)$

4=number of variables

In what complexity class is this query?

F

L (why?)

AC^0 (why?)

Discussion

- The query complexity of $O(n^4)$ if we assumed that the query is fixed and only the database is variable
- If we assume that both query and database are variable, then the complexity is different (and much higher)

- Data complexity v.s. Query complexity

Vardi, *The Complexity of Relational Query Languages*, STOC 1982

Query Q, database D

Moshe Vardi
2008 ACM SIGMOD
Codd Innovation Award

- <u>Data complexity</u>:
  fix Q, complexity = f(D)
- <u>Query complexity</u>:
  fix D, complexity = f(Q)
- <u>Combined complexity</u>:
  complexity = f(D,Q)

Data Complexity

Given a query Q in a query language L
what is the complexity of the following problem?

"Given D, compute Q(D)"

- Language design tradeoff
  - High complexity → L can express rich queries
  - Low complexity → L can be implemented efficiently

Conventions

- The complexity is usually defined for a decision problem
  - Hence, we will study only the complexity of Boolean queries
- The complexity usually assumes some encoding of the input
  - Hence we will encode the database instances using a binary representation

Boolean Queries

<u>Definition</u> A *Boolean Query* is a query that returns either true or false

<u>Factoid</u>: the complexity of L is fully determined by its Boolean queries

Boolean Queries

Definition A *Boolean Query* is a query that returns either true or false

Factoid: the complexity of L is fully determined by its Boolean queries

Non-boolean queries

$Q(x,y) = \exists z.R(x,z) \wedge S(z,y)$

Boolean queries:

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

$a,b \in ADom(D)$

Boolean Queries

Definition A *Boolean Query* is a query that returns either true or false

Factoid: the complexity of L is fully determined by its Boolean queries

Non-boolean queries

$Q(x,y) = \exists z.R(x,z) \wedge S(z,y)$

```
SELECT DISTINCT R.x, S.y
FROM R, S
WHERE R.z = S.z
```

Boolean queries:

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

$a,b \in ADom(D)$

```
SELECT DISTINCT 'yes'
FROM R, S
WHERE R.x='a' and R.y = S.y and S.y='b'
```

Boolean Queries

Definition A *Boolean Query* is a query that returns either true or false

Factoid: the complexity of L is fully determined by its Boolean queries

Non-boolean queries

$Q(x,y) = \exists z.R(x,z) \wedge S(z,y)$

```
SELECT DISTINCT R.x, S.y
FROM R, S
WHERE R.z = S.z
```

$Q(x)=\exists y.R(x,y) \wedge (\forall z.S(y,z) \rightarrow \exists u.R(z,u))$

Boolean queries:

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

$a,b \in ADom(D)$

```
SELECT DISTINCT 'yes'
FROM R, S
WHERE R.x='a' and R.y = S.y and S.y='b'
```

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

Boolean Queries

Definition A *Boolean Query* is a query that returns either true or false

Factoid: the complexity of L is fully determined by its Boolean queries

Non-boolean queries

$Q(x,y) = \exists z.R(x,z) \wedge S(z,y)$

```
SELECT DISTINCT R.x, S.y
FROM R, S
WHERE R.z = S.z
```

$Q(x)=\exists y.R(x,y) \wedge (\forall z.S(y,z) \rightarrow \exists u.R(z,u))$

```
T(x,y) :- R(x,y)
T(x,y) :- T(x,z),R(z,y)
```

Transitive closure

Boolean queries:

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

a,b $\in$ ADom(D)

```
SELECT DISTINCT 'yes'
FROM R, S
WHERE R.x='a' and R.y = S.y and S.y='b'
```

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

```
T(x,y) :- R(x,y)
T(x,y) :- T(x,z),R(z,y)
Answer() :- T('a','b')
```

Reachability

Database Encoding

- We could encode a database D as a list of encodings of tuples, similar to how database systems store data on disk:
  - "[R(a,b),R(cac,bbfg),R(cac,b)],[S(b),S(cac)]"

- Instead, we use a simpler representation, generalizing adjacency matrix for graphs

Database Encoding

Encode **D** = (D, R_1^D, …, R_k^D) as follows:

- Let n = |ADom(D)|
- If R_i has arity k, then encode it as a string of n^k bits:
  - 0 means element $(a_1,\ldots, a_k) \notin R_i^D$
  - 1 means element $(a_1,\ldots, a_k) \in R_i^D$

a	b
a	c
b	b
b	c
c	a

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

The Data Complexity

Fix any Boolean query Q in the query language. Determine the complexity of the following problem:

- Given an input database instance **D** = (D, R_1^D, …, R_k^D), check if Q(**D**) = true.
- This is also known as the *Model Checking Problem*: check if **D** is a model for Q.

What we will discuss next time

- Relational queries
- Datalog and stratified datalog$^{\neg}$
- Datalog$^{\neg}$ with inflationary fixpoint
- Datalog$^{\neg}$ with partial fixpoint

- AC^0
- L = a.k.a. LOGSPACE
- NL= a.k.a. NLOGSPACE
- NC
- P = a.k.a. PTIME
- NP
- PSPACE