Efficiently Querying Contradictory and Uncertain Genealogical Data

Lars E. Olson and David W. Embley
DEG Lab
BYU Computer Science Dept.

Supported by National Science Foundation
Grant #0083127

Introduction

- Integrating data from multiple sources
- Some data just doesn't fit the data model
  - Multiple data sources → conflicting data
  - Uncertain or imprecise data
  - Data that violates constraints
- Sometimes it's not possible to resolve the data
- PAF / Gedcom

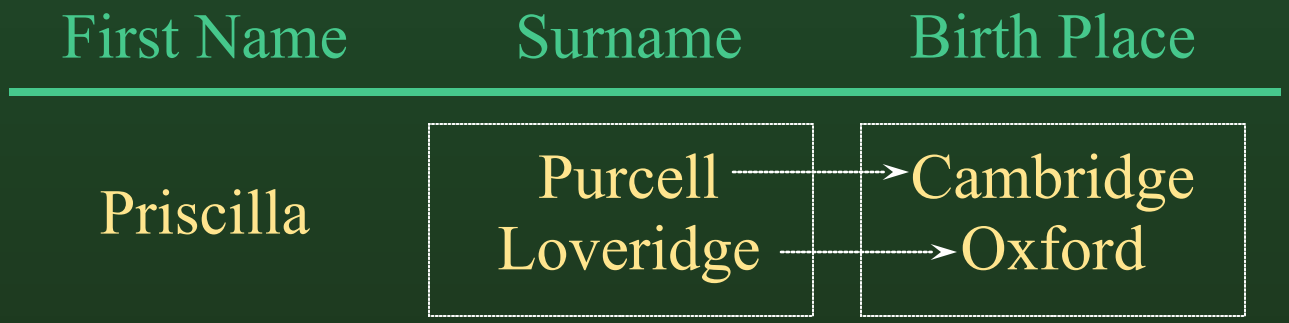
Disjunctive Databases

"OR-tables," Imielinski and Vadaparty, 1989

Name	Birth Date	Marriage Date	Death Date
James I	Dec. 1394	2 Feb. 1423 2 Feb. 1424	21 Feb. 1436 21 Feb. 1437
Joseph Harrison	26 Jan. 1781 26 Jan. 1782 26 Jul. 1782	19 Dec. 1811	5 Apr. 1861
⋮	⋮	⋮	⋮

Shortcomings of “OR-tables”

- Can’t correlate between possible values

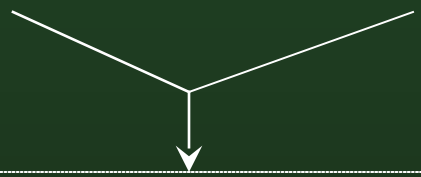


- Answering queries in general is CoNP-complete (Imielinski & Vadaparty)

Sub-relation Data Construct

- Solution: store the correlated data in its own relation

First Name	Surname	Birth Place
Priscilla		



Surname	Birth Place
Purcell	Cambridge
Loveridge	Oxford

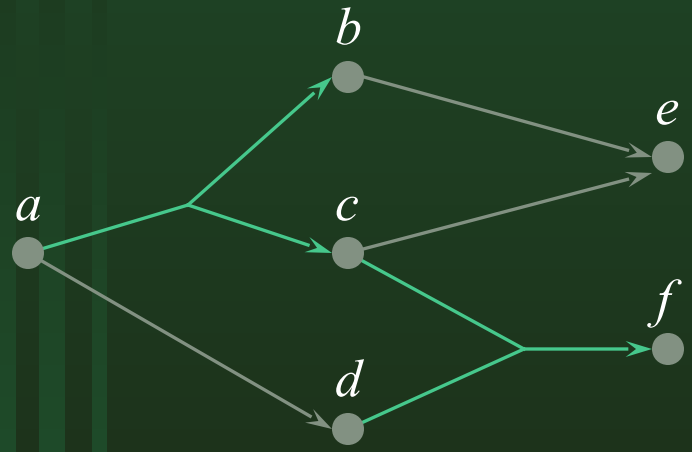
Disjunctive Database Problems

- How do we avoid the CoNP-completeness problem and answer queries efficiently?
- If more than one value is possible, which one is the most likely?
- Other questions to be solved:
  - Where are the constraint violations?
  - How do we map sub-relations to physical storage?
  - How do we efficiently update the database?

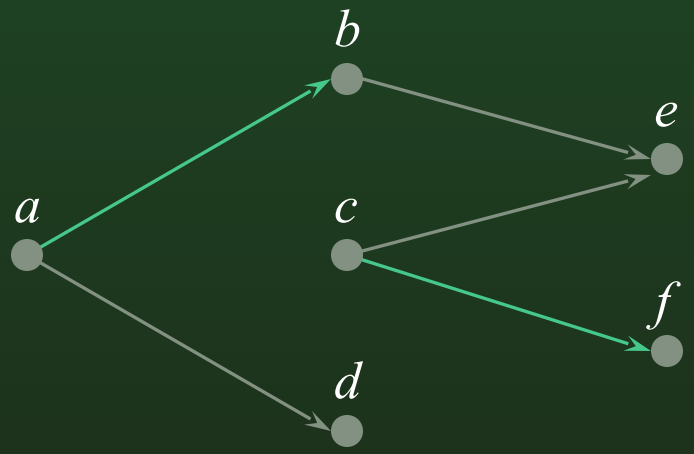
Transitive Closure of Disjunctive Graphs

Solving the CoNP-completeness problem [LYY95]

Disjunctive graph



Possible interpretation



Transitive closure of a: $\{a, d, e\}$

Using Disjunctive Graphs to Answer Queries

Table *Person*:

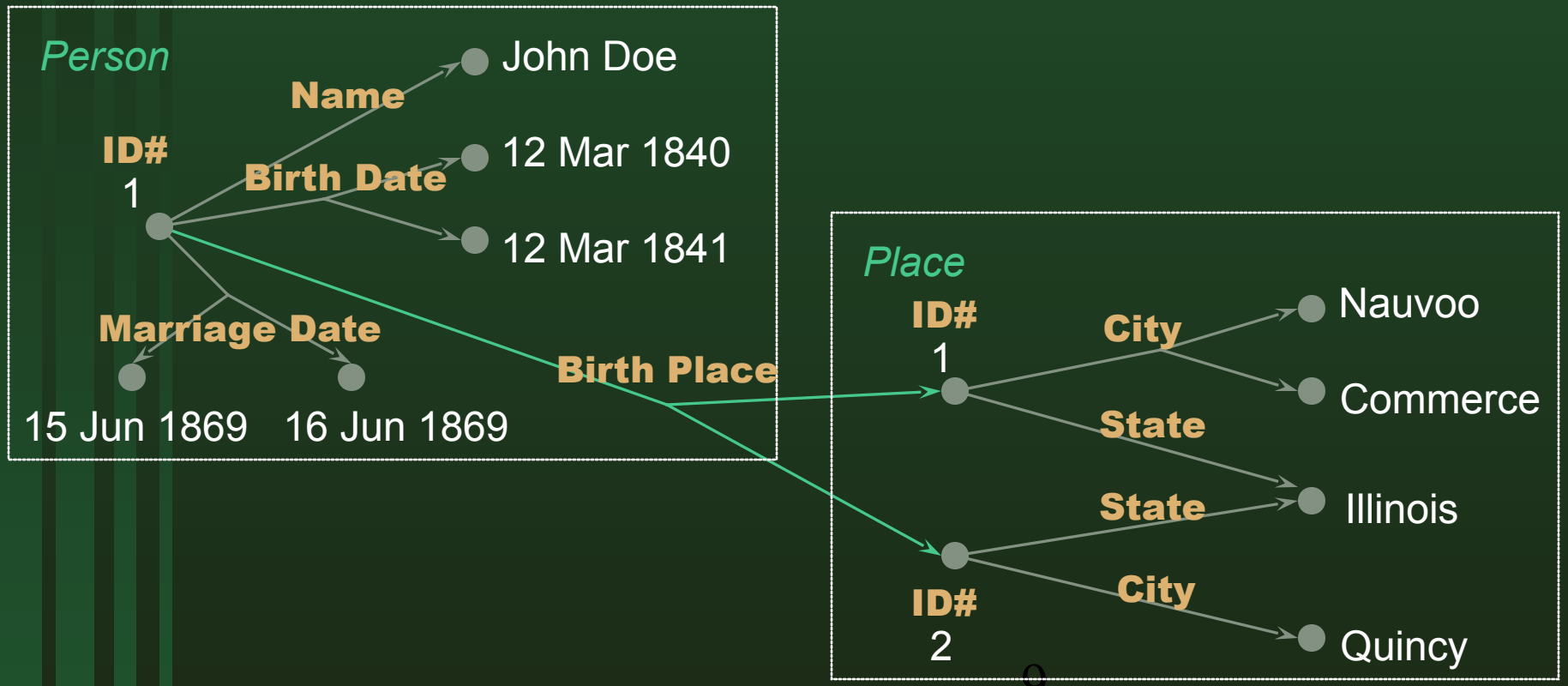
ID#	Name	Birth Date	Birth Place ID# (references Table *Place*)	Marriage Date
1	John Doe	12 Mar. 1840 or 12 Mar. 1841	1 or 2	15 Jun. 1869 or 16 Jun. 1869
⋮	⋮	⋮	⋮	⋮

Table *Place*:

ID#	City	State
1	Commerce or Nauvoo	Illinois
2	Quincy	Illinois
⋮	⋮	⋮

Using Disjunctive Graphs to Answer Queries

$$\pi_{State}(\sigma_{ID=1}Person \bowtie Place)$$

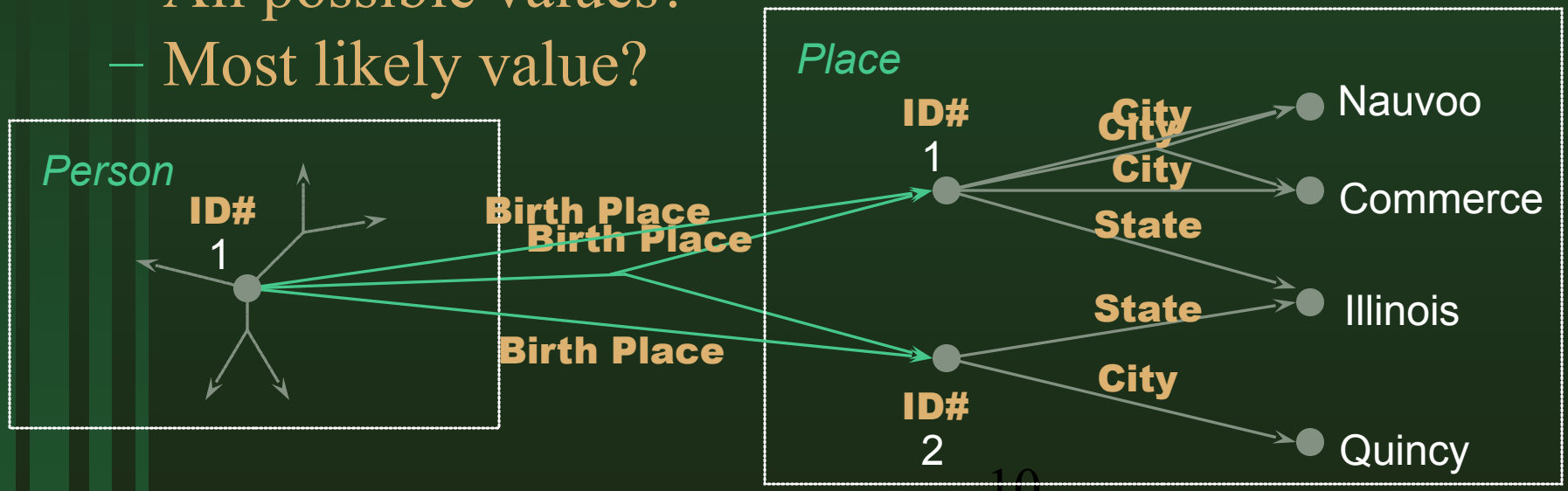


Using Disjunctive Graphs to Answer Queries

$$\pi_{City,State}(\sigma_{ID=1}Person \bowtie Place)$$

…meaning what?

- Definitely known?
- All possible values?
- Most likely value?

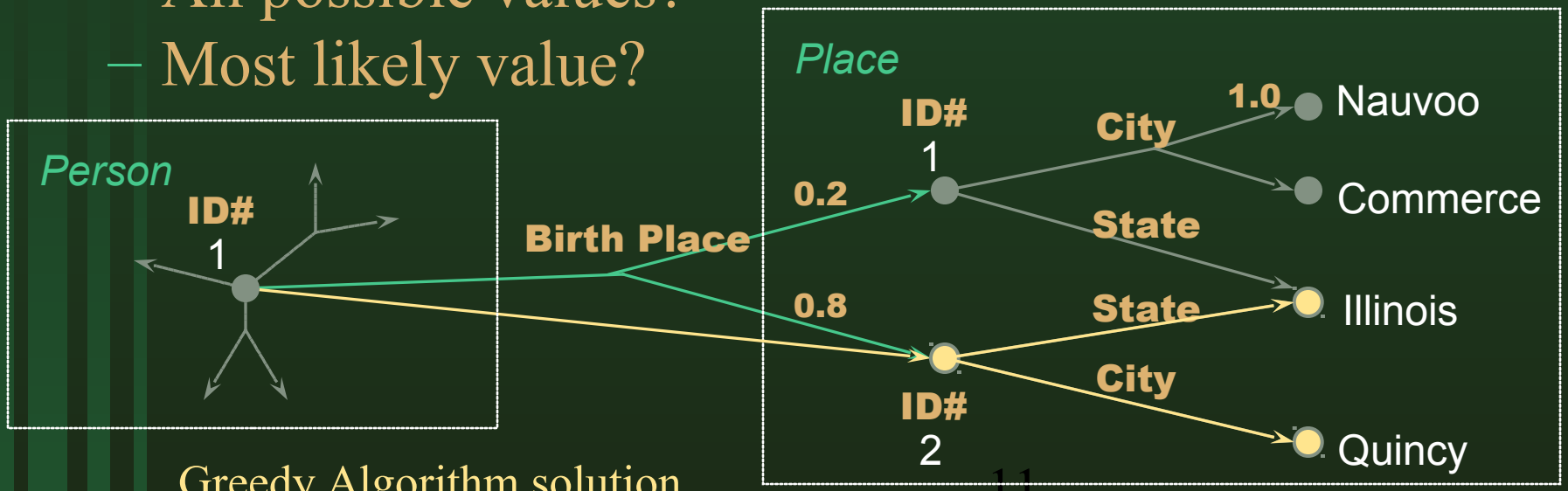


Using Disjunctive Graphs to Answer Queries

$$\pi_{City,State}(\sigma_{ID=1}Person \bowtie Place)$$

…meaning what?

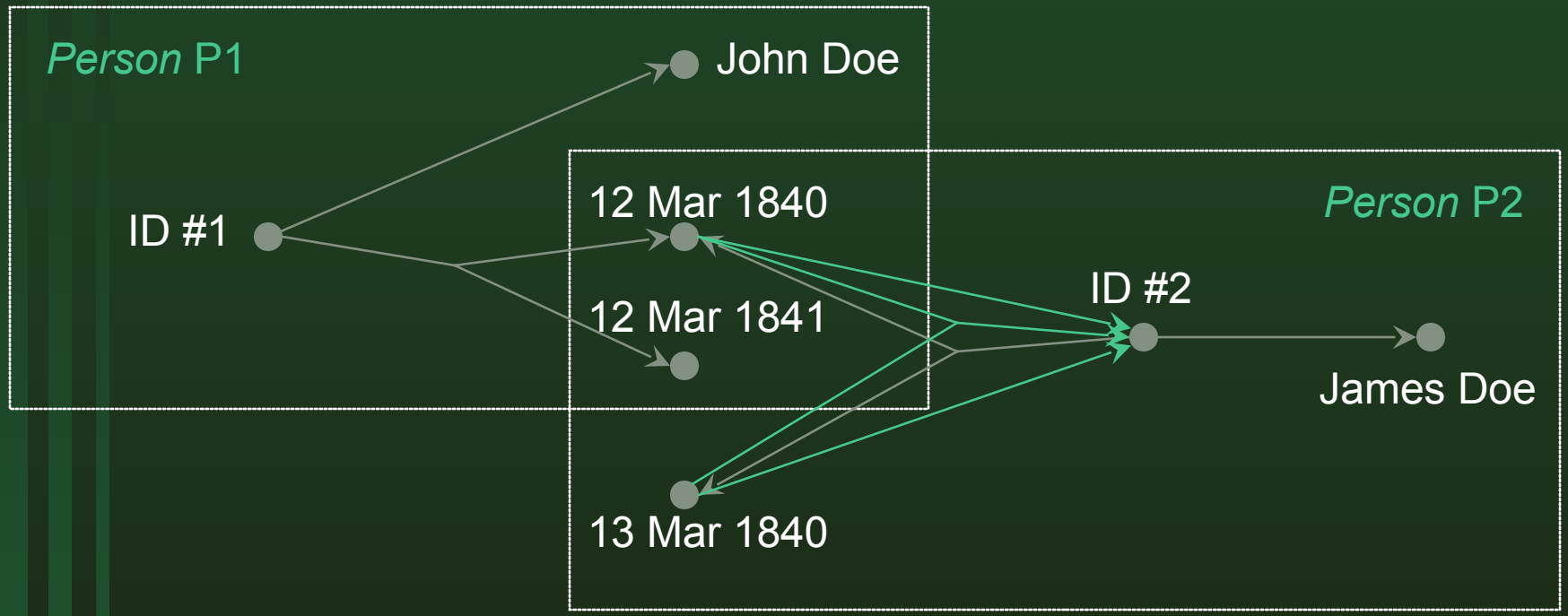
- Definitely known?
- All possible values?
- Most likely value?



Greedy Algorithm solution

Using Disjunctive Graphs to Answer Queries

$$\pi_{\text{P1.Name, P2.Name}}(\text{Person P1} \bowtie_{\text{P1.BirthDate = P2.BirthDate}} \text{Person P2})$$



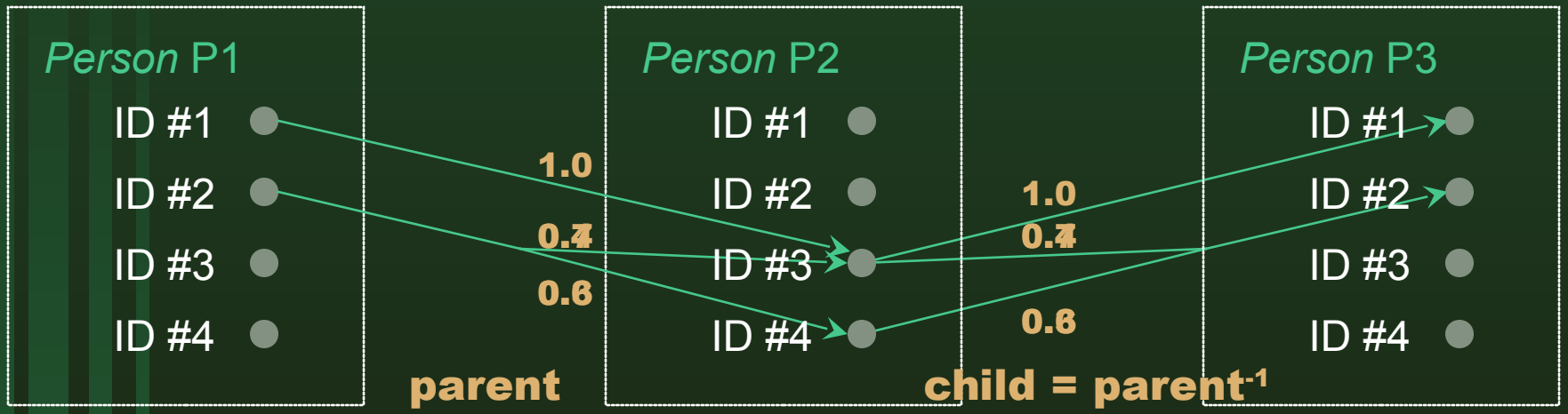
Limiting the Search Space

- In genealogy, most disjunctions are mutually independent
- Disjunctions that aren't independent are limited to immediate family relations
- Build a relation containing all immediate family members

$$(\text{Person P1} \ltimes_{\text{P1.parent = P2.ID}} \text{Person P2} \ltimes_{\text{P2.ID = P3.parent}} \text{Person P3})$$

Limiting the Search Space

- Example constraints:
  - Each parent should be born before their children
  - Each child should be born at least 9 months apart (except multiple births)



Conclusions

- Genealogical data can be stored in a disjunctive database format.
- Many common queries can be computed in polynomial time.
- We can detect intractable queries and limit the search space required, usually enough to get polynomial time.