Combining Consistency and Confidentiality Requirements in First-Order Databases

Lena Wiese

Fakultät für Informatik
LS 6 (ISSI)

September 7–9, 2009

technische universität dortmund

Outline

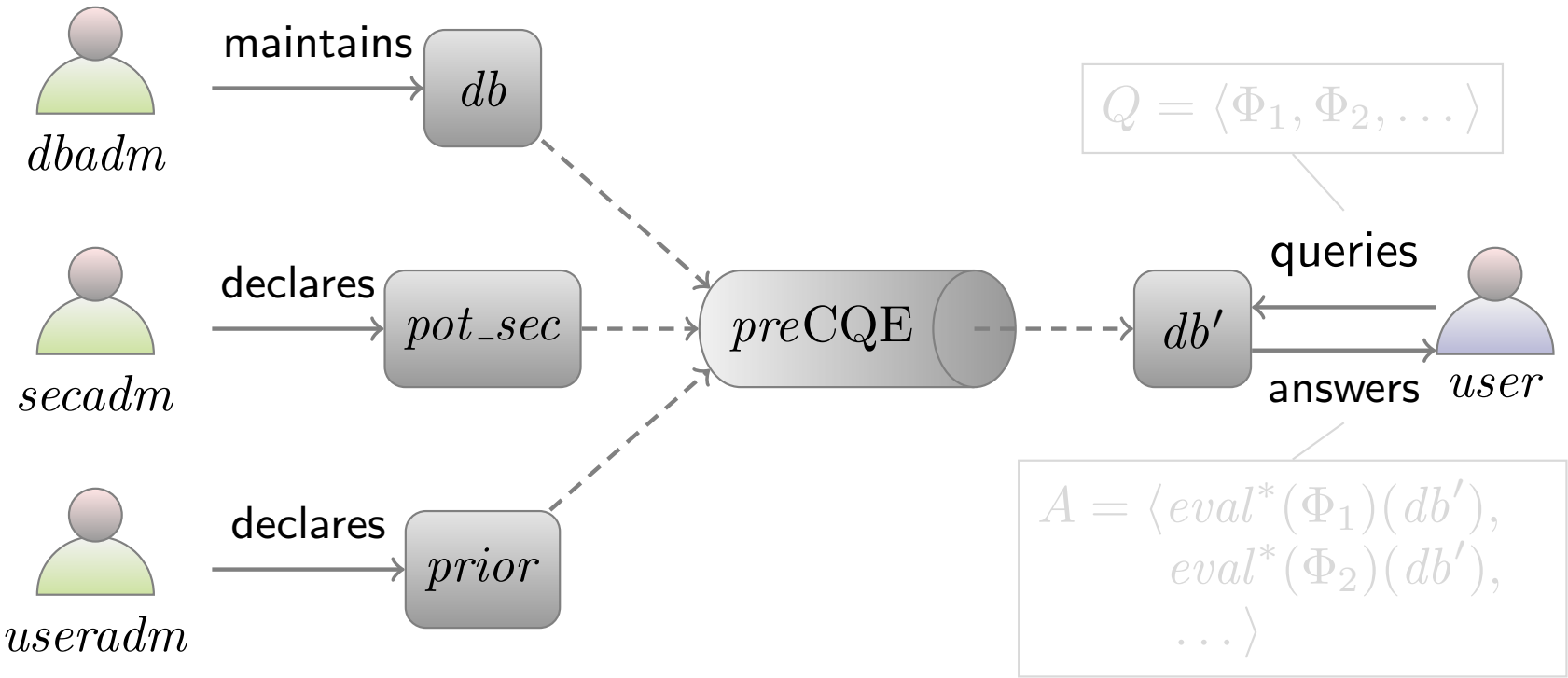
Inference control

- Protect confidential and private information in database instance db
- Personalized security (confidentiality) policy pot_sec
- User profile (a priori knowledge) $prior$
- IC system automatically distorts some answers
  - Avoids harmful user inferences
- Here: modify input database
  - Remove tuples (like Data Privacy; Stouppa/Studer, 2009)
  - Add tuples (like Cover Stories; eg. Galinovic et al, 2007)
- Automatically generate "inference-proof" output instance
- Also related to Data Exchange (eg. Fagin et al, 2005) and Consistent Query Answering (eg. Chomicki, 2007)

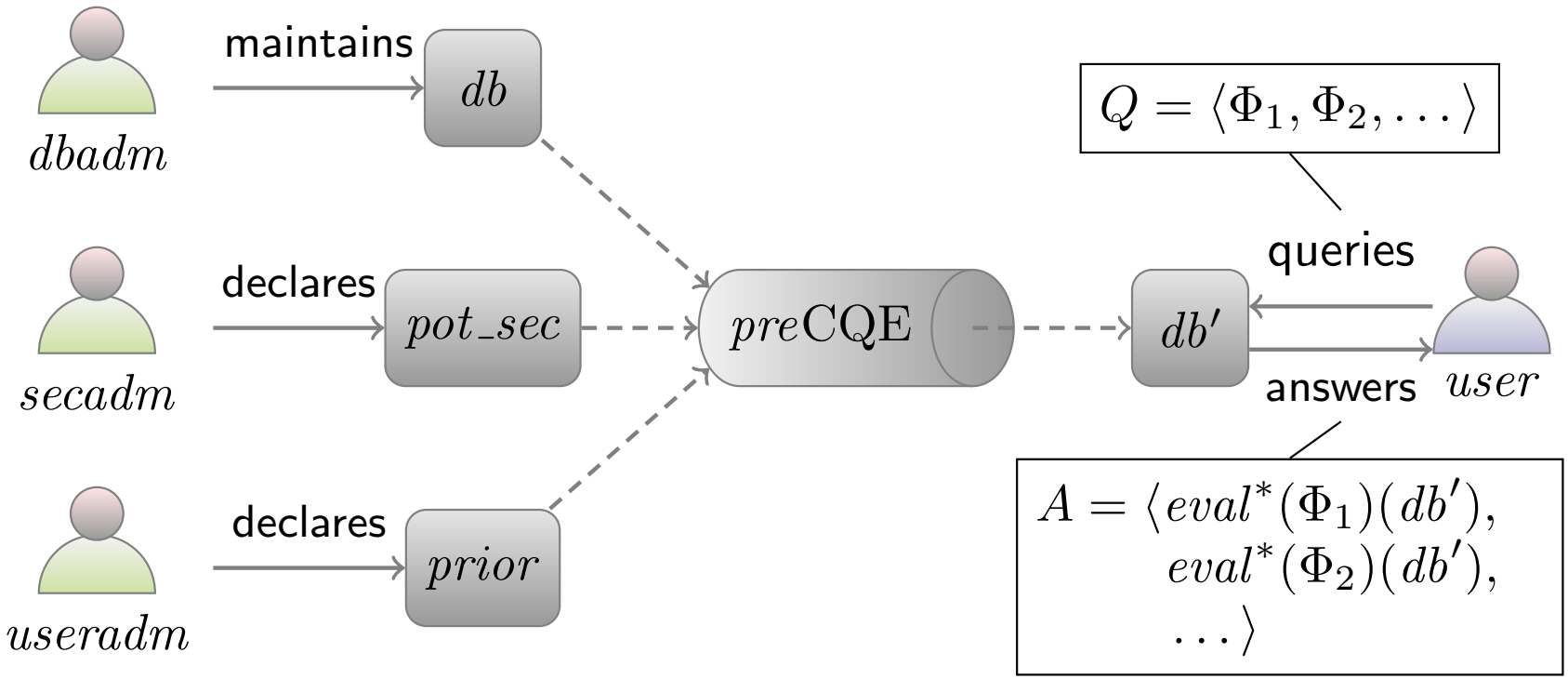
Prior work: Controlled Query Evaluation (CQE)

- (Biskup/Bonatti, 2007; Biskup/Wiese, 2008)...
- Logical view of relational data model
- Database schema $DS = \langle \mathcal{P}, \mathcal{D} \rangle$ with relation names $\mathcal{P}$ and database dependencies $\mathcal{D}$
- Infinite domain of values (constants) dom
- Complete database instance as finite set of tuples (ground atoms) + closed world assumption
- Relational calculus as query language

*pre*CQE: Preprocessing for CQE



$preCQE$: Preprocessing for CQE



Preprocessing for CQE

Definition: Inference-proofness of db'

1. [Consistency] $I^{db'} \models prior$
2. [Confidentiality] $I^{db'} \not\models \Psi$ for every $\Psi \in pot\_sec$

Definition: Distortion distance (amount of modified tuples)

[Availability] $db\_dist(db') \quad := \quad card((db \setminus db') \cup (db' \setminus db))$

- Find db' that satisfies constraint set $C := prior \cup Neg(pot\_sec)$
- Minimize amount of modified tuples db_dist (maximize availability)
- No impact on runtime performance
- No user history (log file) has to be stored

Example

$$\begin{aligned} \mathcal{P} &= \{Ill, Treat\}, \\ dom &= \{\text{Pete}, \text{Mary}, \text{Lisa}, \text{Paul}, \ldots, \\ &\quad \text{Aids}, \text{Flu}, \text{Cancer}, \text{Myopia}, \ldots \\ &\quad \text{MedA}, \text{MedB}, \text{MedC}, \ldots\} \end{aligned}$$

db:

Ill	$Name$	$Diagnosis$
	Pete	Aids
	Mary	Cancer

$Treat$	$Name$	$Treatment$
	Pete	MedA
	Mary	MedB

$$\begin{aligned} prior &= \{\forall x(Treat(x, \text{MedA}) \rightarrow Ill(x, \text{Aids}) \vee Ill(x, \text{Cancer})), \\ &\quad \forall x(Treat(x, \text{MedB}) \rightarrow Ill(x, \text{Cancer}) \vee Ill(x, \text{Flu}))\} \\ pot\_sec &= \{\exists x Ill(x, \text{Aids}), \exists x Ill(x, \text{Cancer})\} \end{aligned}$$

Example

$$
\begin{aligned}
prior &= \{\forall x(\mathit{Treat}(x, \mathsf{MedA}) \rightarrow \mathit{Ill}(x, \mathsf{Aids}) \vee \mathit{Ill}(x, \mathsf{Cancer})),\\
&\quad\ \forall x(\mathit{Treat}(x, \mathsf{MedB}) \rightarrow \mathit{Ill}(x, \mathsf{Cancer}) \vee \mathit{Ill}(x, \mathsf{Flu}))\}\\
pot\_sec &= \{\exists x \mathit{Ill}(x, \mathsf{Aids}), \exists x \mathit{Ill}(x, \mathsf{Cancer})\}\\
Neg(pot\_sec) &= \{\forall x \neg \mathit{Ill}(x, \mathsf{Aids}), \forall x \neg \mathit{Ill}(x, \mathsf{Cancer})\}
\end{aligned}
$$

Constraint set $C := prior \cup Neg(pot\_sec)$

db:

Ill	Name	$\mathit{Diagnosis}$
	Pete	Aids
	Mary	Cancer

Treat	Name	$\mathit{Treatment}$
	Pete	MedA
	Mary	MedB

Example

$$
\begin{aligned}
prior &= \{\forall x(\mathit{Treat}(x, \mathsf{MedA}) \rightarrow \mathit{Ill}(x, \mathsf{Aids}) \vee \mathit{Ill}(x, \mathsf{Cancer})),\\
&\quad\ \forall x(\mathit{Treat}(x, \mathsf{MedB}) \rightarrow \mathit{Ill}(x, \mathsf{Cancer}) \vee \mathit{Ill}(x, \mathsf{Flu}))\}\\
pot\_sec &= \{\exists x \mathit{Ill}(x, \mathsf{Aids}), \exists x \mathit{Ill}(x, \mathsf{Cancer})\}\\
Neg(pot\_sec) &= \{\forall x \neg \mathit{Ill}(x, \mathsf{Aids}), \forall x \neg \mathit{Ill}(x, \mathsf{Cancer})\}
\end{aligned}
$$

Constraint set $C := prior \cup Neg(pot\_sec)$

db_1':

Ill	*Name*	*Diagnosis*
	Pete	Aids
	Mary	Cancer

Treat	*Name*	*Treatment*
	Pete	MedA
	Mary	MedB

$$db\_dist(db_1') = 4$$

Example

$$
\begin{aligned}
prior &= \{\forall x(Treat(x, \mathsf{MedA}) \rightarrow Ill(x, \mathsf{Aids}) \vee Ill(x, \mathsf{Cancer})),\\
&\quad\ \forall x(Treat(x, \mathsf{MedB}) \rightarrow Ill(x, \mathsf{Cancer}) \vee Ill(x, \mathsf{Flu}))\}\\
pot\_sec &= \{\exists x Ill(x, \mathsf{Aids}), \exists x Ill(x, \mathsf{Cancer})\}\\
Neg(pot\_sec) &= \{\forall x \neg Ill(x, \mathsf{Aids}), \forall x \neg Ill(x, \mathsf{Cancer})\}
\end{aligned}
$$

Constraint set $C := prior \cup Neg(pot\_sec)$

db_2':

Ill	$Name$	$Diagnosis$
	Pete	Aids
	Mary	Cancer
	Mary	Flu

$Treat$	$Name$	$Treatment$
	Pete	MedA
	Mary	MedB

$$db\_dist(db_2') = db\_dist(db_1') = 4$$

Outline

Undecidability of satisfiability problem

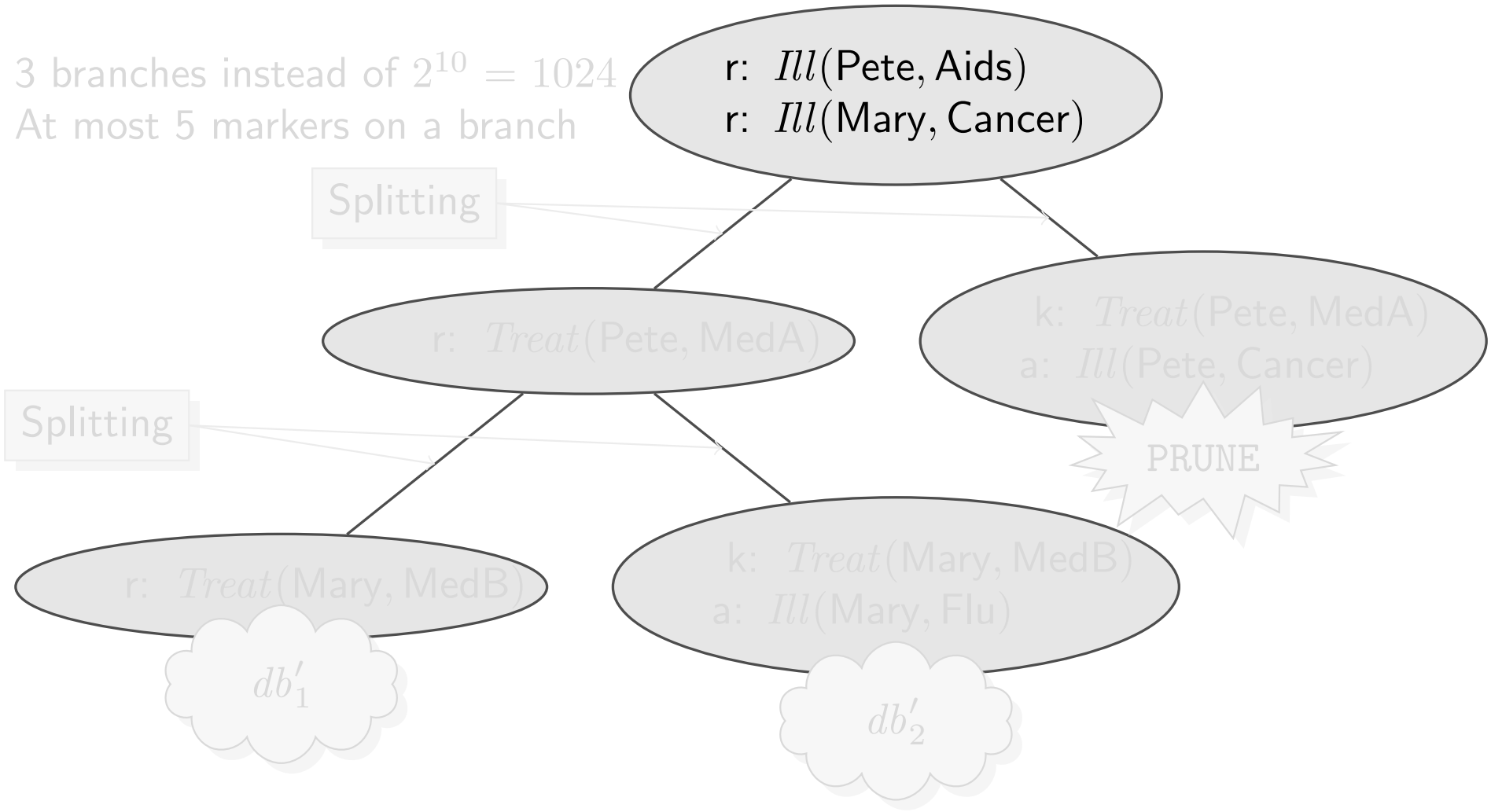
- For complete database, known potential secrets & data modification, find db' such that $I^{db'} \models C$ and $db\_dist(db') \longrightarrow \min$
- Undecidability of satisfiability problem for predicate logic (reproduced in (Börger et al, 2001))
- Identify syntactical restrictions for constraint set C to make problem decidable
- "Allowed universal formulas" in prenex literal normal form
  - Subset of "allowed formulas" (Van Gelder/Topor, 1991)
  - "Active domain" semantics, $adom$: constants in db and C

$preCQE$ subprocedures

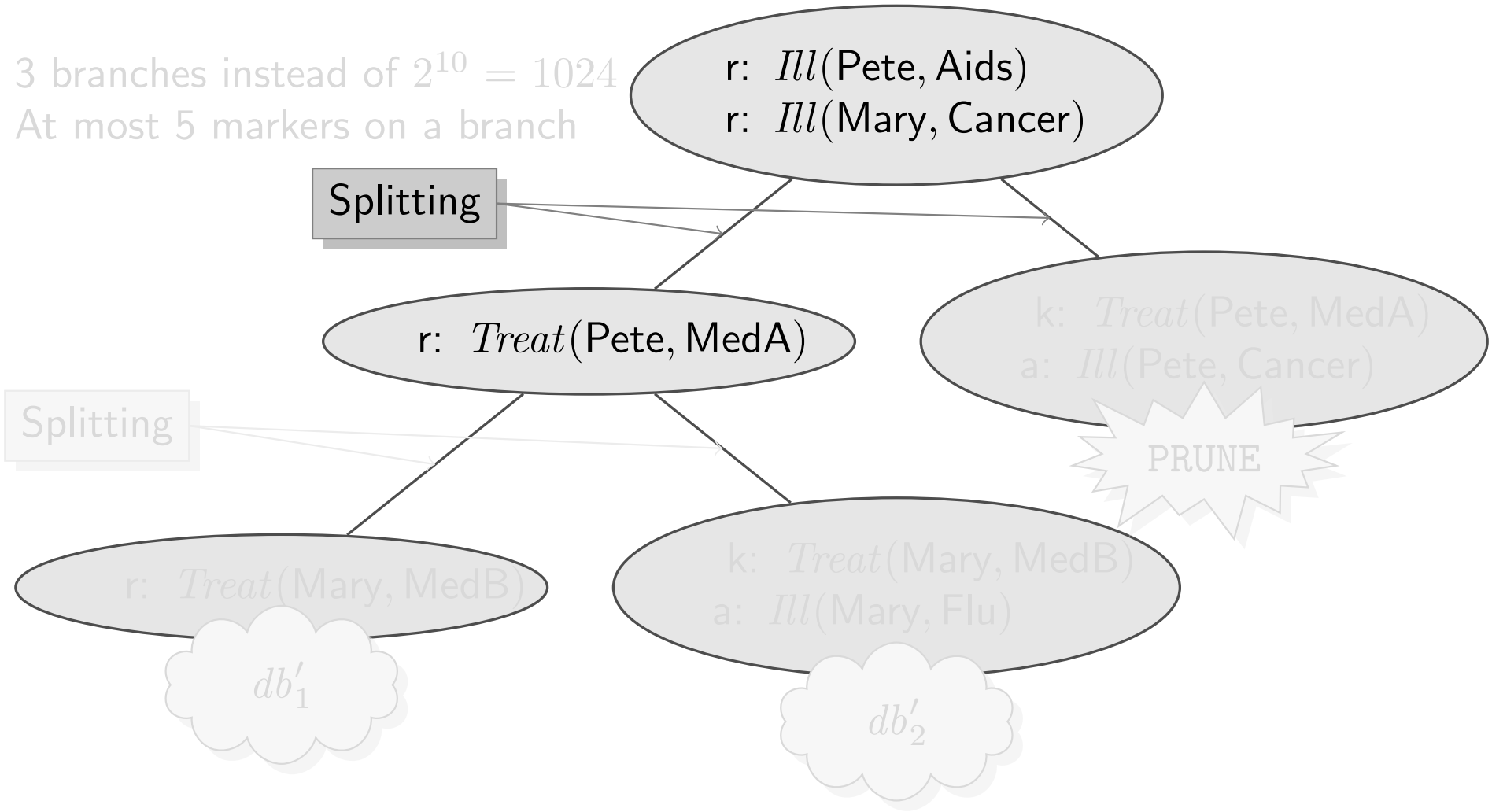
Branch-and-Bound depth-first search tree

- `INIT`
- `GROUND`
  - find relevant ground instantiations for universal quantifiers
- `SIMP`
- `SPLIT`
  - try two truth values for ground atom
- `MARK`
  - mark ground atom as k(ept), a(dded), r(emoved) or l(eft out)

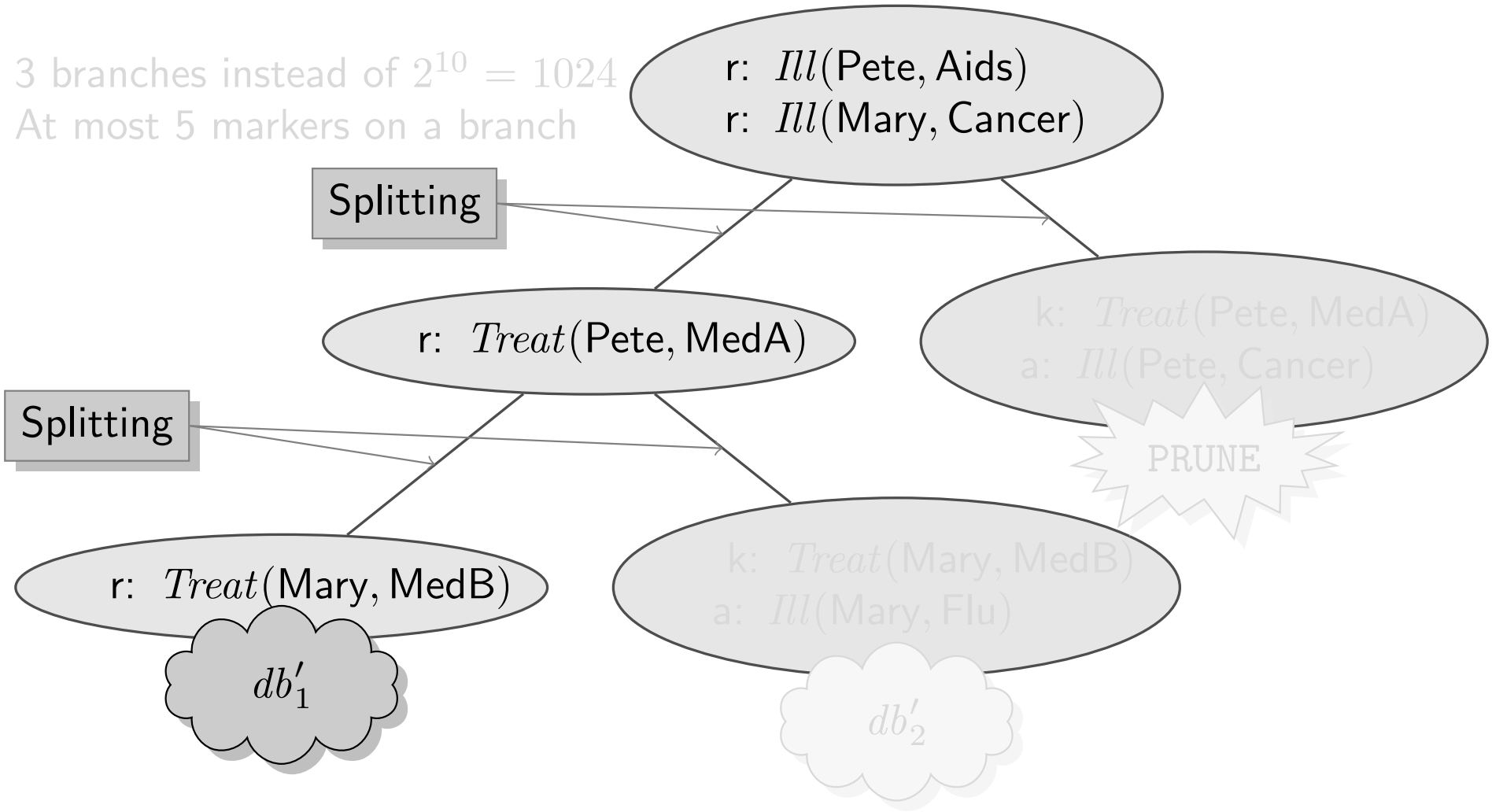
Example: $preCQE$ search tree



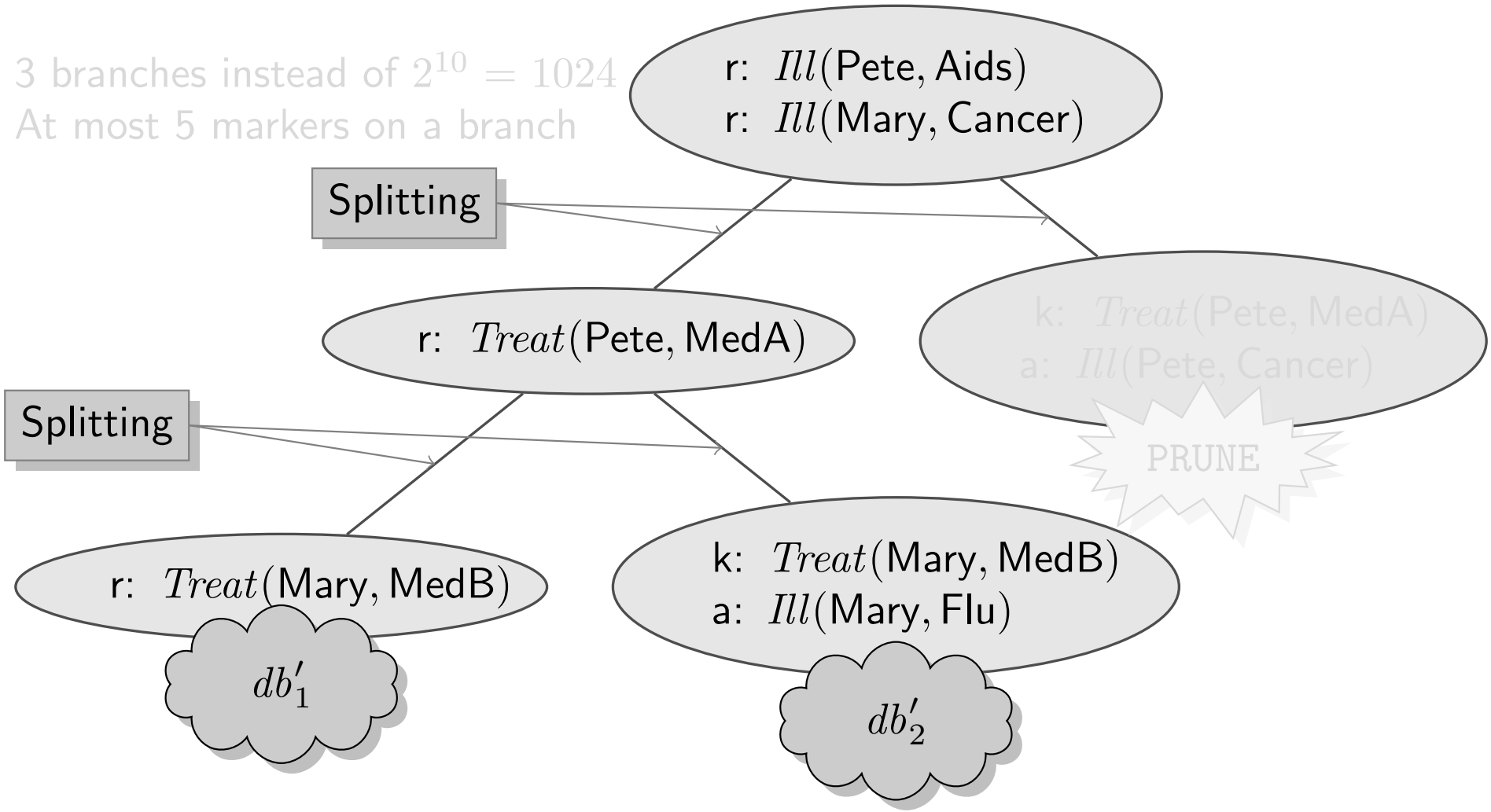
Example: $preCQE$ search tree



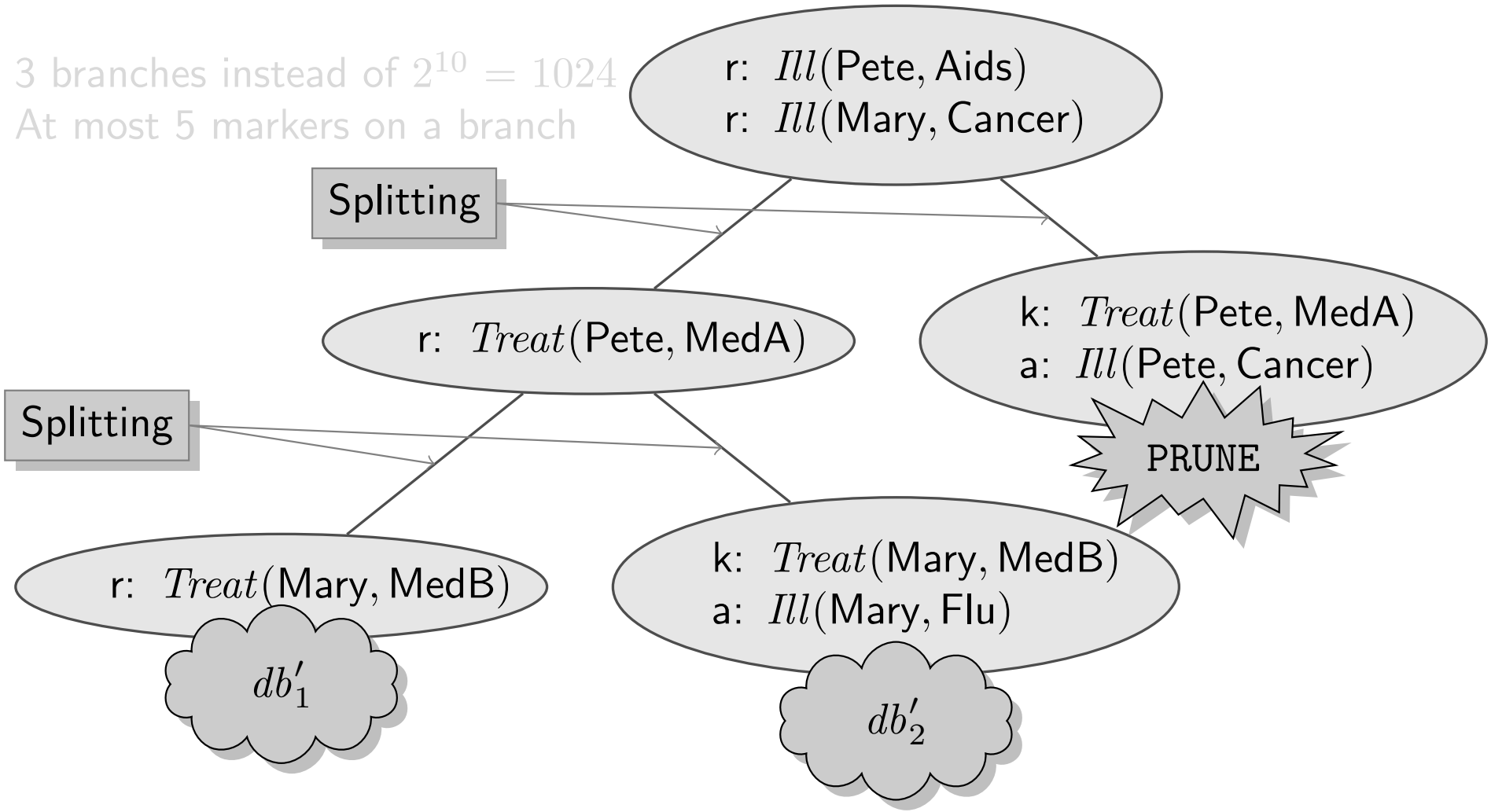
Example: $preCQE$ search tree



Example: $preCQE$ search tree

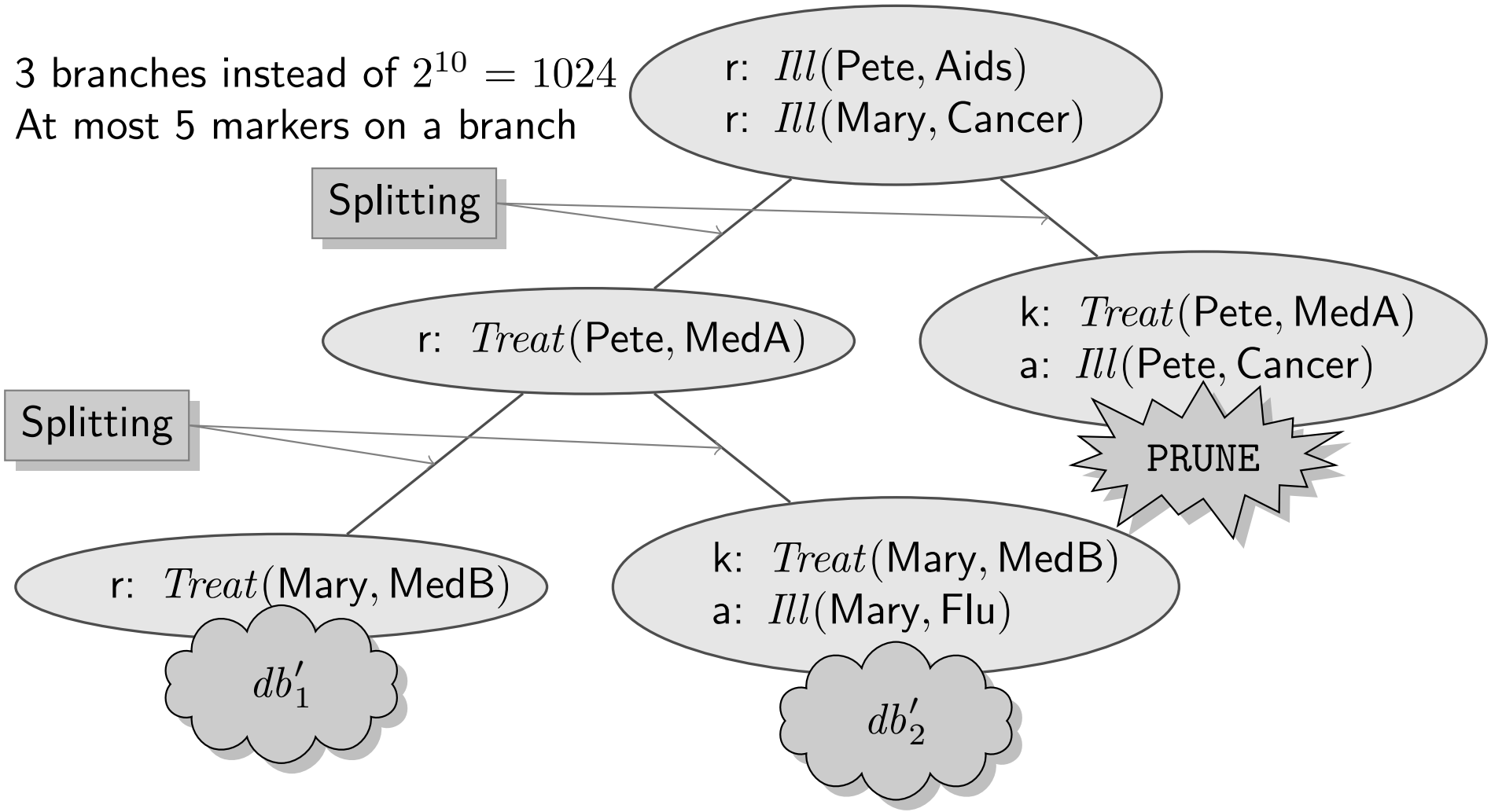


Example: $preCQE$ search tree



Example: $preCQE$ search tree

3 branches instead of $2^{10} = 1024$
At most 5 markers on a branch



Key results

Theorem: Termination of $preCQE$

For a set C of allowed universal constraints, $preCQE$ terminates in a finite amount of time

Proof:

- At most k different ground atoms with $adom$ constants where

$$k := \sum_{\substack{P \in \mathcal{P} \\ P \text{ occurs in } C}} card(adom)^{arity(P)}$$

- At most 2^k branches in the search tree

Key results

Theorem: Satisfiability soundness of $preCQE$

For a set C of allowed universal constraints, if db' is a database instance, it is inference-proof (hence, a model of C)

Proof:

- No violated constraints left

Corollary: Refutation completeness of $preCQE$

For a set C of allowed universal constraints, if C is unsatisfiable, db' is undefined

Key results

Theorem: Satisfiability soundness of $preCQE$

For a set C of allowed universal constraints, if db' is a database instance, it is inference-proof (hence, a model of C)

Proof:

- No violated constraints left

Corollary: Refutation completeness of $preCQE$

For a set C of allowed universal constraints, if C is unsatisfiable, db' is undefined

Key results

Theorem: Refutation soundness of $preCQE$

For a set C of allowed universal constraints, if db' is undefined, then C is unsatisfiable

- Not trivial because of efficiency of $preCQE$
- Not all $adom$-ground atoms explicitly handled
  1. Only violated constraints and affected ground atoms are considered
  2. If a truth assignment is unequivocal, ground atoms are marked directly without splitting
  3. Branches are pruned if a better solution has already been found
  4. Branches are pruned as soon as a conflict occurs
- $preCQE$ search tree in best case does not contain all possible 2^k branches

Proof of refutation soundness

Herbrand's Theorem (Herbrand, 1930; here as in Cook/Nguyen, 2009)

Let S be a set of closed universal formulas. Then S is unsatisfiable iff some finite set S_0 of ground instances of formulas in S is propositionally unsatisfiable

Herbrand's Theorem with semantic tree (Chang/Lee, 1973; for clauses)

Let S be a set of closed universal formulas. Then S is unsatisfiable iff for some finite set S_0 of ground instances of formulas in S there is a closed semantic tree

- Construct semantic tree out of $preCQE$ search tree
- Identify set C_0 of ground instances
- Show that semantic tree is closed for C_0

Key results

Corollary: Satisfiability completeness of $preCQE$

For a set C of allowed universal constraints, if C is satisfiable, db' is a database instance

Theorem: Distortion minimality of solution

For a set C of allowed universal constraints, if $preCQE$ finds a solution db', then it is distortion-minimal

- For other constraints (existential or weakly acyclic fragments) similar:
  - Depth and width of $preCQE$ search tree is bounded
  - Fix mapping of existentially quantified variables to invented constants for refutation soundness

Key results

Corollary: Satisfiability completeness of $preCQE$

For a set C of allowed universal constraints, if C is satisfiable, db' is a database instance

Theorem: Distortion minimality of solution

For a set C of allowed universal constraints, if $preCQE$ finds a solution db', then it is distortion-minimal

- For other constraints (existential or weakly acyclic fragments) similar:
  - Depth and width of $preCQE$ search tree is bounded
  - Fix mapping of existentially quantified variables to invented constants for refutation soundness

Key results

Corollary: Satisfiability completeness of $preCQE$

For a set C of allowed universal constraints, if C is satisfiable, db' is a database instance

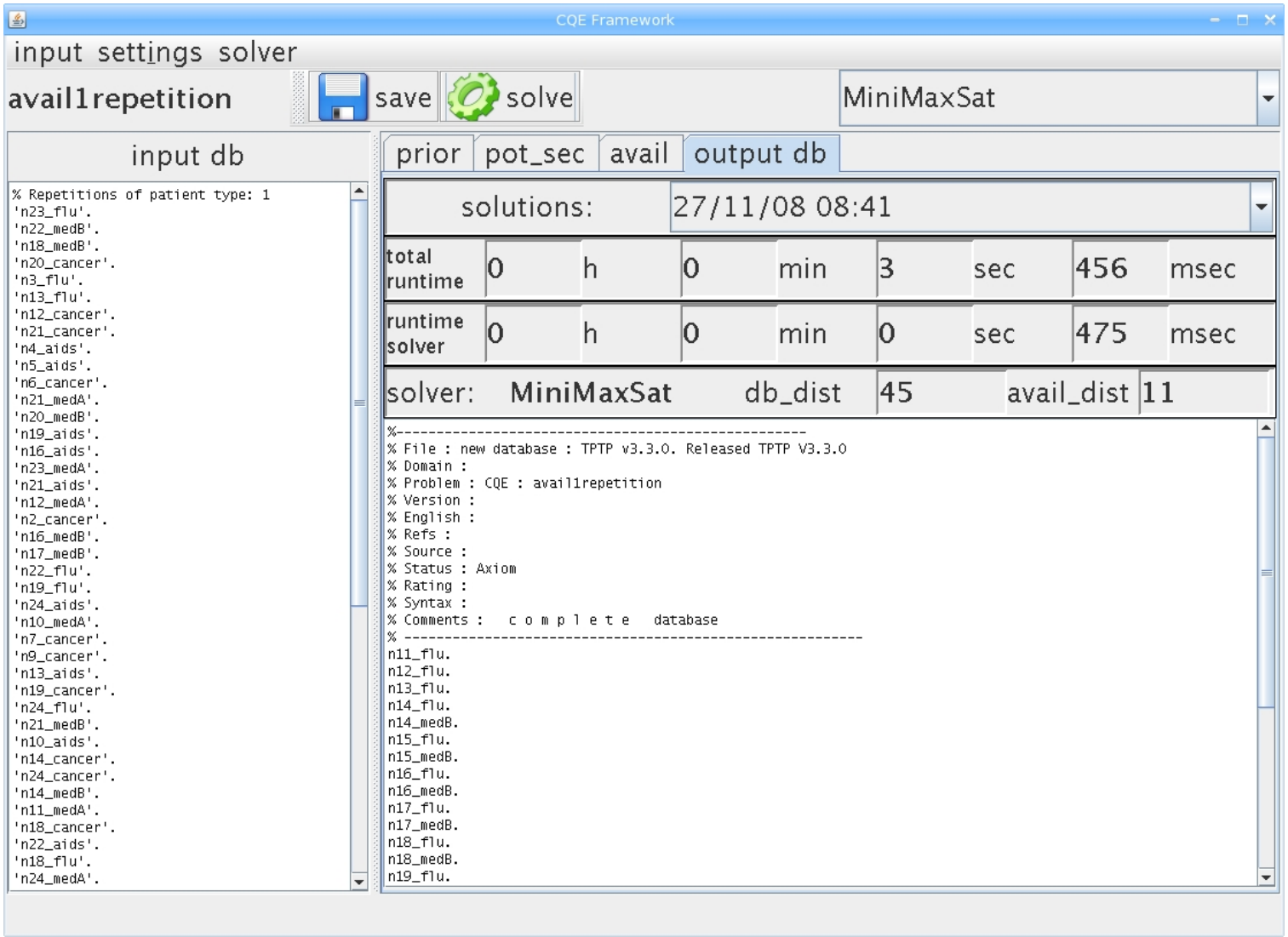
Theorem: Distortion minimality of solution

For a set C of allowed universal constraints, if $preCQE$ finds a solution db', then it is distortion-minimal

- For other constraints (existential or weakly acyclic fragments) similar:
  - Depth and width of $preCQE$ search tree is bounded
  - Fix mapping of existentially quantified variables to invented constants for refutation soundness

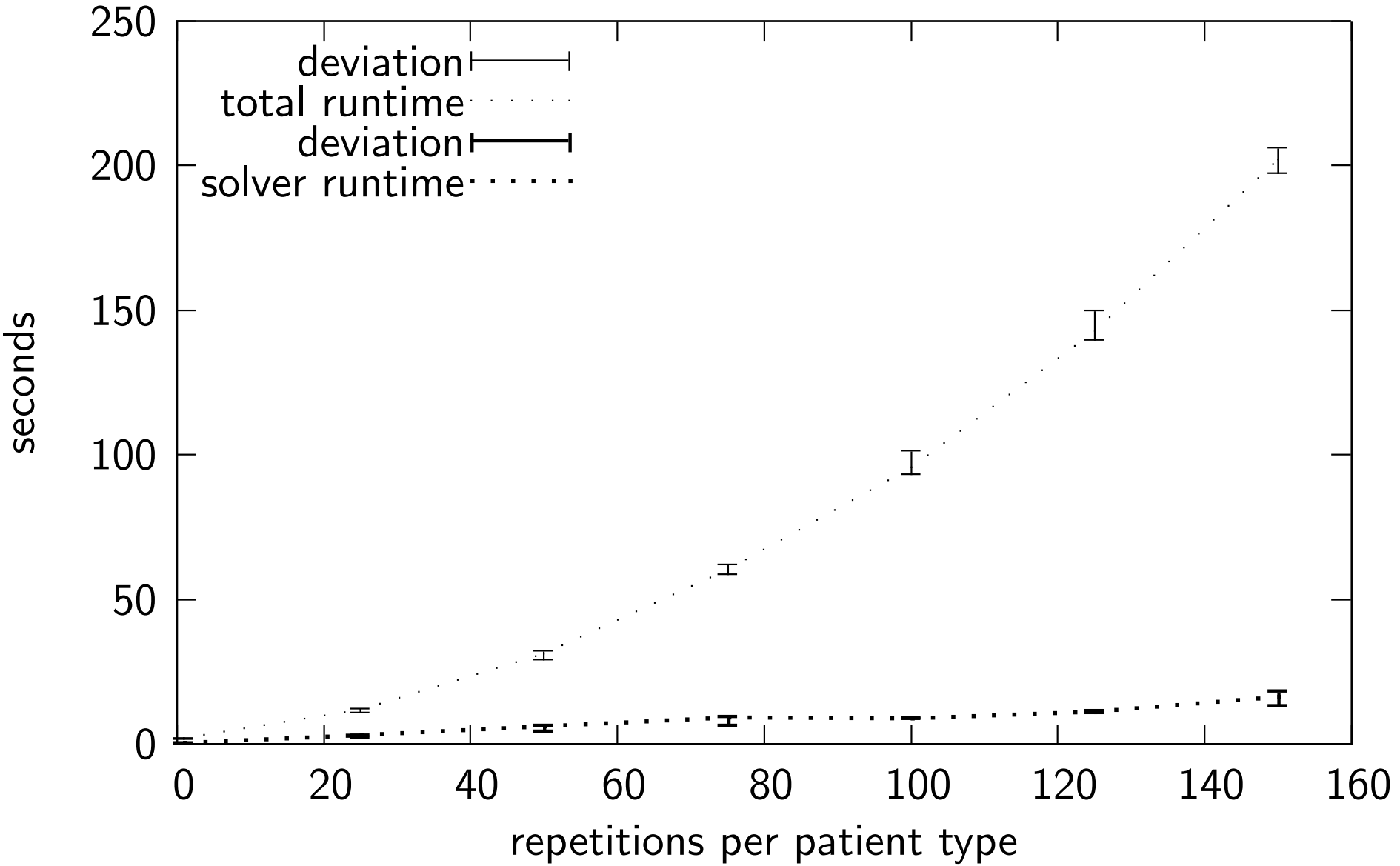
Outline

User interface



Average test results

rep.	avg. msec total	avg. msec solver	dec. vars.	clauses soft	clauses hard
1	1930	184	120	120	96
25	11974	3092	3000	3000	2400
50	31304	6135	6000	6000	4800
75	60459	8991	9000	9000	7200
100	95792	8902	12000	12000	9600
125	142843	11171	15000	15000	12000
150	202067	16429	18000	18000	16800


deviation
total runtime
deviation
solver runtime
seconds
repetitions per patient type
0
50
100
150
200
250
0
20
40
60
80
100
120
140
160

Outline

Achievements

- Consistency with $prior$, confidentiality of pot_sec, maximal availability of unmodified tuples with db_dist
- Unique combination of model generation and distance minimization in infinite domain
- Quantifier handling without a need to expand them into ground conjunctions or disjunctions
- Only minor restrictions on the syntax of constraint formulas
- Both addition and deletion of tuples as modification primitives
- Optimized for complete databases with efficient query evaluation function
- Output in form of complete database instance
- Termination, soundness and completeness for appropriate fragments