Foundations of Query Languages

Dr. Fang Wei

Lehrstuhl für Datenbanken und Informationssysteme
Universität Freiburg

SS 2011

A legitimate question

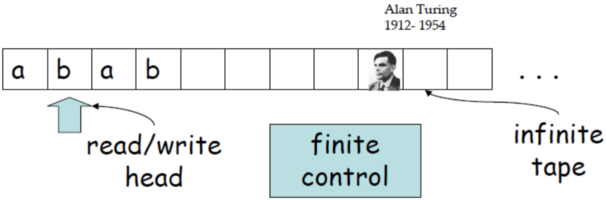
Given a query Q in RA, is there at all adatabase D such that $Q(D) \neq \emptyset$?

- If there is no such database, then the query Q makes no sense and we can directly replace it by the empty result.
- Could save much runtime. Also in the case of subquerie

Today we will show from first principles that this problem is undecidable.

Turing Machine

General model of computation: Turing Machine



Turing Machine

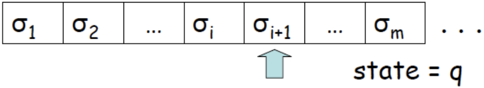
Turing Machine:

$$(Q, \Sigma, \Gamma, \sigma, q_{start}, q_{accept}, q_{reject})$$

Q: set of states (q_{start}, q_1, . . ., q_n, q_{accept}, q_{reject})
Σ: input alphabet ($\{0, 1\}$ suffices)
Γ: tape alphabet ($\Sigma \subseteq \Gamma$), e.g. $\{0, 1\#, t\}$
σ: $Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, -\}$ transition function
$q_{start} \in Q$: start state
$q_{accept} \in Q$: accept state (could also be a state set)
$q_{reject} \in Q$: reject state (could also be a state set)

TM Configurations

Useful convention: Turing Machine configurations.



Any point in computation represented by string:

$$C = \sigma_1\ \sigma_2\ \ldots\ \sigma_i\ q\ \sigma_{i+1}\ \sigma_{i+1} \ldots \sigma_m$$

start configuration for single-tape TM on input x:

$$q_{start}\ x_1\ x_2 \ldots x_n$$

Turing Machine

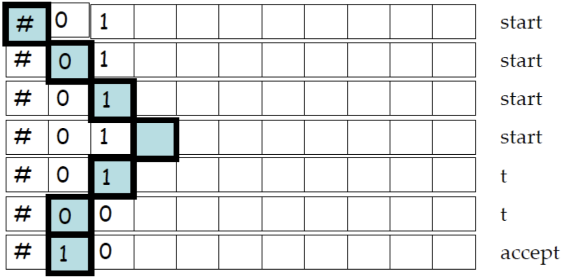
Three notions of computation with Turing machines. In all, input x written on tape

- function computation: output $f(x)$ is left on the tape when TM halts
- language decision: TM halts in state q_{accept} if $x \in L$; TM halts in state q_{reject} if $x \notin L$.
- language acceptance: TM halts in state q_{accept} if $x \in L$; may loop forever otherwise.

Turing Machine Example

q	σ	δ (q, σ)
start	0	(start, 0, R)
start	1	(start, 1, R)
start	⊔	(t, ⊔ , L)
start	#	(start, #, R)

q	σ	δ(q, σ)
t	0	(accept, 1, -)
t	1	(t, 0, L)
t	#	(accept, #, R)



Extended Church Turing Thesis

Consequence of extended Church-Turing Thesis: all reasonable physically realizable models of computation can be efficientlysimulated by a TM

- e.g. multi-tape vs. single tape TM
- e.g. RAM model

Turing Machine

There exist (natural) undecidable problems

$$HALT = \{(M, x) : Mhaltsoninputx\}$$

Theorem

HALT is undecidable, even for single-tape TMs.

Undecidability of Halting

Suppose that TM $H(M, x)$ decides whether $M(x)$ halts.

- Define new TM H' on input M: $H'(M)$
  - if $H(M, M)$ accepts, then $H'(M)$ loops
  - if $H(M, M)$ rejects, then $H'(M)$ halts
- Consider H' on input H' : $H'(H')$
  - if $H'(H')$ halts, then $H(H', H')$ rejects, which means $H'(H')$ loops
  - if $H'(H')$ loops, then $H(H', H')$ accepts, which means $H'(H')$ halts, contradiction.

Trakhtenbrot's Theorem

Trakhtenbrot's Theorem

For every relational vocabulary σ with at least one binary relation symbol, it is undecidable whether an FO sentence ϕ over σ is finitely satisfiable.

– Boris A. Trakhtenbrot: *1921 Brichevo, Belarus; now at Tel Aviv University.

This theorem does the job. Translated into database terminology, it reads:

Undecidability of FO Queries

For a database schema σ with at least one binary relation, it is undecidable, whether a Boolean FO or RA query Q over σ is satisfied by at least one database.

Proof of Trakhtenbrot's Theorem

Proof Idea:

- Define a relational signature σ suitable for encoding finite computations of a TM
- For each specific TM M and input I, transform M into an FO formula $\phi_{M,I}$ such that for each structure (i.e., database) D over σ, we have $D \models \phi_{M,I}$ iff M with input I halts.

Proof of Trakhtenbrot's Theorem

$$(Q, \Sigma, \Gamma, \sigma, q_{start}, q_{accept}, q_{reject})$$

Simplifying assumptions:

- σ may have several unary and binary relations. (We could always encode them into a single binary relation! $\rightarrow$ exercise)
- Tape alphabet of $M : \Gamma = \Sigma = \{0, 1\}$. (Can always be obtained by simple coding tricks, e.g.: 0 $\rightarrow$10; 1$\rightarrow$01; # $\rightarrow$11; t $\rightarrow$ 00)

Proof of Trakhtenbrot's Theorem

Further assumptions:

- The head never moves to the left of the first cell.
- The machine halts iff it enters state q_{accept} or state q_{reject} and it halts only in these states.

$\rightarrow$ These two conditions can be enforced by easy modifications of M that preserve 'halting?equivalence'.

Proof of Trakhtenbrot's Theorem

$$TM : (Q, \Sigma, \Gamma, \sigma, q_{start}, q_{accept}, q_{reject})$$

Relational signature (database schema):

$$\sigma = \{<, Min(.), T_0(.,.), T_1(.,.), H(.,.), S(.,.)\}$$

With the following intended meaning:

- $<$ is a linear order, as usual,we write $x < y$ instead of $<(x, y)$.
- The elements of this linear order will be used to simulate both time instants and tape position (=cell numbers).
- $Min(x)$ is true for the smallest element of $<$ only.
- T_0 and T_1 are tape predicates: $T_0(p, t)$ indicates that cell number p at time t contains 0, ...
- $H(p, t)$ indicates that the head at time t is at position p(i.e., at cell number p)
- $S(s, t)$ indicates that at instant t the machine is in state s.

Proof of Trakhtenbrot's Theorem

$\phi_{M,I}$ is the conjunction of the following sentences:
A sentence stating that $<$ is a linear order and that *Min* contains its minimal element.This has in turn the following conjuncts:

$\forall x, y(x \neq y \rightarrow (x < y \vee y < x))$	*totality*
$\forall x, y \neg(x < y \wedge y < x)$	*antisymmetry* + *antireflexivity*
$\forall x, y, z(x < y \wedge y < z \rightarrow x < z)$	*transitivity*
$\forall x, y(Min(x) \rightarrow (x = y \vee x < y))$	

Proof of Trakhtenbrot's Theorem

A remaining large sentence:

$$\exists s_0, s_1, \ldots, s_k \ (\phi_{states} \wedge \phi_{rest})$$

where s_i is a variable representing state i of TM (we assume the TM has $k+1$ states), and

$$\phi_{states} \equiv \bigwedge_{i \neq j} s_i \neq s_j$$

where ϕ_{rest} further describes the machine behavior as follows.

Proof of Trakhtenbrot's Theorem

ϕ_{rest} contains the following conjuncts:

initial configuration

A formula defining the initial configuration of M with I on its input tape which in turn contains the following conjuncts: Assuming the input string I has length n. Denote its i-th bit by b_i. Then for each input position $0 \leq i < n$ (we start at 0):

$$\forall p, t \ ((Min(t) \wedge [p = i]) \rightarrow T_{b_i}(p, t))$$

where $[p = i]$ is an abbreviation for a FO formula stating that p is the i-th element of ¡

$\rightarrow$ This describes that at instant 0 the tape contains the input string I.

Proof of Trakhtenbrot's Theorem

initial configuration (cont.)

$$\forall p, t \ (([p \geq n] \land Min(t)) \rightarrow T_0(p, t))$$

$\longrightarrow$ all other cells contain 0 at time 0.

$$\forall t \ (Min(t) \rightarrow H(t, t))$$

$\longrightarrow$ the head is initially at the start position 0.

$$\forall t(Min(t) \rightarrow S(s_0, t))$$

$\longrightarrow$ the machine is initially in state 0.

Proof of Trakhtenbrot's Theorem

state formula

in every configuration, each cell of the tape contains exactly one symbol

$$\forall p, t \; ((T_0(p,t) \vee T_1(p,t)) \wedge (T_0(p,t) \not\equiv T_1(p,t)))$$

state formula (2)

at any time the machine is in exactly one state

$$\forall t((\bigvee_{1 \leq i \leq k} S(s_i, t)) \wedge \bigwedge_{i \neq j} \neg(S(s_i, t) \wedge S(s_j, t)))$$

state formula (3)

at any time the head is at exactly one position ($\rightarrow$ exercise)

Proof of Trakhtenbrot's Theorem

state transition

In particular, for each transition tuple of the transition relation σ one formula. For instance, if a transition specifies that when the machine is in state 4 and reads 0 it writes 1, moves to the right and switches to state 6, we will express this as:

$$\begin{aligned}\forall p, t\ ((H(p,t) \wedge T_0(p,t) \wedge S(s_4,t)) \rightarrow \\ \exists p', t'(p' = p+1 \wedge t' = t+1 \wedge H(p',t') \wedge S(s_6,t') \wedge T_1(p,t') \wedge \\ \forall r \neq p(T_0(r,t') \equiv T_0(r,t)))\end{aligned}$$

Proof of Trakhtenbrot's Theorem

halting condition

We must say that M halts on input I: assume $q_{accept} = a$ and $q_{reject} = b$.

$$\exists t \; (S(s_a, t) \vee S(s_b, t))$$

This completes the description of $\phi_{M,I}$. This formula faithfully describes M on Input I, thus M halts on input I if there exists a database D: $D \models \phi_{M,I}$. QED

Further undecidability results

The following problems are undecidable:

- Safety of a FO query (i.e., domain independence).
- Equivalence of two FO (or RA) queries
- Query containment $Q_1 \subseteq Q_2$. (Recall that this means: $\forall D Q_1(D) \subseteq Q_2(D)$.

Corollary to Trakhtenbrot's Theorem

For a database schema σ with at least one binary relation, it is undecidable, whether an SQL query Q over σ will produce a non-empty result on at least one database.

Thus, there is no algorithm for perfect SQL optimization.