Alternation, Space Complexity and QFPAbit

Alternation

- $$ATIME(f(n)) = \{L \mid L \textit{ is decidable by an } O(f(n)) \textit{ Alternating Turing Machine}\}$$
- $$APTIME = \bigcup_{k \in \mathbb{N}} ATIME(n^k)$$

Alternation

- Clearly NP$\subseteq$APTIME
- co-NP$\subseteq$APTIME
- Example: $TQBF_2$ is neither in NP nor in co-NP but it is in APTIME

Space Complexity

- A Turing Machine is said to be f(n) space if for an input of length n it scans only f(n) cells.
- $$SPACE(f(n)) = \{L \mid L \text{ is decidable by an } O(f(n)) \text{ space Turing Machine}\}$$
- $$PSPACE = \bigcup_{k \in \mathbb{N}} SPACE(n^k)$$
- Interesting result: NPSPACE=PSPACE (Savitch)

APTIME vs PSPACE

- Theorem: For $f(n) \geq n$ we have $ATIME(f(n)) \subseteq SPACE(f(n)) \subseteq ATIME(f^2(n))$
- Corollary: APTIME=PSPACE

Polynomial Hierarchy

- Let i be a natural number. A Σ_i-Alternating Turing Machine is an ATM that changes between performing existential and universal steps at most i-1 times, starting with existential steps.
- A Π_i-alternating Turing Machine is defined just like a Σ_i-ATM but starts with universal steps.

Polynomial Hierarchy

- $\Sigma_i P = \bigcup_k \Sigma_i TIME(n^k)$
- $\Pi_i P = \bigcup_k \Pi_i TIME(n^k)$
- For example, Σ_1P=NP and Π_1P=co-NP.

QF Presburger Arithmetic

- Satisfiability NP-hard: propositional SAT can be reduced to it
- Satisfiability also in NP (see references)

QFPAbit

- We add bitwise operators $\bar{\wedge}$, $\bar{\vee}$, $\bar{\neg}$ as functions. They are interpreted as acting on the Two's Complement Binary Representation of the numbers
- Note that the "+" isn't actually necessary anymore (see homework)

Two's Complement Encoding

- $< b_k, b_{k-1}, ..., b_0 >= -b_k 2^k + \Sigma_{i=0}^{k-1} b_i 2^i$

QFPAbit stronger than QFPA

- The set of numbers that satisfy a QFPA formula is **ultimately periodic**
- But consider

$$pow2(x) \equiv 1 + ((x-1)\bar{\nabla}x) = 2x \wedge x > 0$$

- x satisfies pow2 if and only if x is a power of 2. The set of all powers of 2 is not ultimately periodic.

QFPAbit Satisfiability

- Given a QFPAbit formula, does there exist an assignment to the variables such that the formula is satisfied?
- Is in PSPACE
- Clearly NP-hard
- Proven to be co-NP-hard
- Conjectured to be PSPACE-complete

QFPAbitSAT∈PSPACE

- We will reduce a given QFPAbit formula to an *Alternating Finite Automaton*
- AFA Emptiness ∈ PSPACE

Alternating Finite Automata

- An AFA is a tuple (Q,V,δ,I,F) where
  - $Q=\{q_1,q_2,..q_m\}$ is the set of state variables
  - $V=\{v_1,v_2...v_n\}$ is the set of input variables
  - $\delta:Q\rightarrow Prop(Q\cup V)$ is the transition function
  - $I\in Prop(Q\cup V)$ is the initial formula
  - $F:Q\rightarrow\{0,1\}$ is the final function that maps states to boolean values

Alternating Finite Automata

- The alphabet of the automaton defined on the previous slide is $\{0,1\}^n$
- An AFA should be viewed more like a "formula constructing device"
- Note that these automata can't accept the empty string
- An AFA can be regarded as a description of a DFA accepting the language where every word is reversed

Boolean operations

- Given an AFA $A_1 = (Q_1, V, \delta_1, I_1, F_1)$ and an AFA $A_2 = (Q_2, V, \delta_2, I_2, F_2)$, define:
  - $\neg A_1 = (Q_1, V, \delta_1, \neg I_1, F_1)$
  - $A_1 \wedge A_2 = (Q_1 \cup Q_2, V, \delta_1 \cup \delta_2, I_1 \wedge I_2, F_1 \cup F_2)$
  - $A_1 \vee A_2 = (Q_1 \cup Q_2, V, \delta_1 \cup \delta_2, I_1 \vee I_2, F_1 \cup F_2)$

Boolean operations

- It is easy to see that the following holds:
  - $L(\neg A_1) = L^c(A_1) \setminus \{\epsilon\}$
  - $L(A_1 \wedge A_2) = L(A_1) \cap (A_2)$
  - $L(A_1 \vee A_2) = L(A_1) \cup (A_2)$

References

- M Sipser: Introduction to The Theory of Computation (2nd ed.); Thomson Course Technology 2006
- T Schuele, K Schneider: Verification of Data Paths Using Unbounded Integers: Automata Strike Back
- S Seshia, R Bryant: Deciding quantifier-free Presburger formulas using parameterized solution bounds; Logical Methods in Computer Science 1(2:6) (2005) 1–26