Computational Logic

Implementations of Herbrand's Theorem

Damiano Zanardini

UPM European Master in Computational Logic (EMCL)
School of Computer Science
Technical University of Madrid
damiano@fi.upm.es

Academic Year 2008/2009

Introduction

General idea

- generate incrementally sets S_i of ground instances by going through the *levels* $H_0, H_1, .., H_k, ..$ of the Herbrand Universe (*level-saturation*)
- for every set S_i, transform it in order to find a *contradiction*, i.e, to prove that it is unsatisfiable
- relies on the *contradiction lemma*

Generation

- the technique used for checking $SAT(S)$ is independent of the technique for generating S
- we can suppose that all methods presented in this section generate S in the same way (with level-saturation)

Complexity

- note that deciding $SAT(S)$ is the well-known $\mathcal{NP}$-complete SAT problem

Introduction

Lemma (contradiction)

A formula F is unsatisfiable iff it is possible to derive a contradiction from it:
$[F] \vdash G \wedge \neg G$

Proof.

❶ $[F] \vdash G \wedge \neg G$ iff $\vdash F \rightarrow G \wedge \neg G$ (deduction theorem)

❷ $\vdash F \rightarrow G \wedge \neg G$ iff, for every interpretation, (1) $I(F) = \mathbf{f}$; or (2) $I(F) = \mathbf{t}$ and $I(G \wedge \neg G) = \mathbf{t}$

❸ $I(G \wedge \neg G) = \mathbf{f}$ for every I, so that $\vdash F \rightarrow G \wedge \neg G$ iff $I(F) = \mathbf{f}$ for every I

❹ $\vdash F \rightarrow G \wedge \neg G$ iff F is unsatisfiable (by ❸)

❺ $[F] \vdash G \wedge \neg G$ iff F is unsatisfiable (by ❶ and ❹)

Gilmore's method

Method: for a set of clauses $\mathcal{C}$

$i = 0;$
$S = \emptyset;$
while $(SAT(S))$
  $H_i =$ the i-th level of $H(\mathcal{C})$
  $X = \{C' \mid C \in \mathcal{C}$ and C' is obtained from C
    by replacing variables with terms in $H_i\};$
  $S = S \cup X;$
  $i = i + 1;$

Satisfiability

- a method for verifying $SAT(S)$ is needed
- Gilmore chose one: *multiplication*

Gilmore's method

Multiplication

- put S in Disjunctive Normal Form ($DNF(S)$)
  - disjunction of conjunctions of literals, ex. $(p \wedge q) \vee r \vee (q \wedge \neg r)$
- search for a contradiction in every conjunction

a: if the contradiction is found *everywhere*, then the set is unsatisfiable

b: if there exists a conjunct which does not contain a contradiction (see lemma Gil-1), then the set is satisfiable

Lemma (Gil-1)

Given a conjunction F of propositions, a contradiction can be derived iff it is a subformula of F

Lemma ($DNF(F)$)

For every (quantifier-free) formula F, $DNF(F)$ exists and is equivalent to F

Gilmore's method

Theorem

A propositional formula F is unsatisfiable iff $DNF(F)$ contains a contradiction in every conjuncts

Proof.

1. F is unsatisfiable iff $DNF(F)$ is (Lemma $DNF(F)$)
2. $DNF(F) = D_1 \vee .. \vee D_n$ is unsatisfiable iff we can derive a contradiction from it (contradiction lemma)
3. $DNF(F)$ is unsatisfiable iff every D_i (conjunction of literals) is
4. $DNF(F)$ is unsatisfiable iff every D_i contains a contradiction (Lemma Gil-1)
4. F is unsatisfiable iff every D_i of $DNF(F)$ contains a contradiction (by ❶ and ❹)

The method of Davis-Putnam

General idea

To simplify the set S of ground instances, getting a new set S' by means of four *rules*, in order to make the detection of a *contradiction* easier

The rules

1. tautology rule
2. one-literal rule
3. pure-literal rule
4. splitting rule

The method of Davis-Putnam

1. Tautology rule

Given a set of ground instances, delete all instances which are *tautologies*

Example

$$\begin{array}{rcl} S & = & \{p,\ q,\ r \vee \neg r\} \\ S' & = & \{p,\ q\} \end{array}$$

clearly, S is satisfiable iff S' is

Lemma (tautology rule)

Since tautologies are always true, eliminating them does not affect satisfiability: the remaining set S' is satisfiable iff S is

The method of Davis-Putnam

2 One-literal rule

If there is a *unit* instance L in S (i.e., a ground instance which only consists of the literal L), then S' can be obtained iteratively by

- deleting all instances in S which contain L
- deleting $\neg L$ *from* the instances in S which contain $\neg L$

Example

$$\begin{array}{rcll}
S & = & \{ \neg p \vee \neg u,\ p \vee q \vee \neg r,\ p \vee \neg q,\ \neg p,\ r,\ u \} & \rightsquigarrow \text{[rule on } \neg p] \\
 & & \{ q \vee \neg r,\ \neg q,\ r,\ u \} & \rightsquigarrow \text{[rule on } \neg q] \\
 & & \{ \neg r,\ r,\ u \} & \rightsquigarrow \text{[rule on } \neg r] \\
S' & = & \{ \Box,\ u \} &
\end{array}$$

the *empty clause* $\Box$ (which can be obtained from r or $\neg r$) means that there is a contradiction: S' is unsatisfiable (like S)

The method of Davis-Putnam

Lemma (one-literal rule)

$S = \{L,\ (L \vee F_1), .., (L \vee F_n),\ (\neg L \vee G_1), .., (\neg L \vee G_m),\ H_1, .., H_p\}$ is unsatisfiable iff $S' = \{G_1, .., G_m,\ H_1, .., H_p\}$ is

- *provided neither L nor $\neg L$ occur in any H_k*

Proof ($\rightarrow$).

1. S is unsatisfiable
2. suppose $\{G_1, .., G_m,\ H_1, .., H_p\}$ is not: then, there exists an interpretation $\mathcal{I}$ which makes all G_j and H_k true
3. if $\mathcal{I}$ also verifies L (it is always possible to find such $\mathcal{I}$), then it verifies all $L \vee F_i$, so that it satisfies the original set
4. contradiction ❷, ❸: $\{G_1, .., G_m,\ H_1, .., H_p\}$ is unsatisfiable

The method of Davis-Putnam

Lemma (one-literal rule)

$S = \{L,\ (L \vee F_1), .., \ (L \vee F_n),\ (\neg L \vee G_1), .., \ (\neg L \vee G_m),\ H_1, .., \ H_p\}$ is unsatisfiable iff $S' = \{G_1, .., \ G_m,\ H_1, .., \ H_p\}$ is

- *provided neither L nor $\neg L$ occur in any H_k*

Proof ($\leftarrow$).

❶ $\{G_1, .., \ G_m,\ H_1, .., \ H_p\}$ is unsatisfiable

❷ suppose S is not: then, there exists an interpretation $\mathcal{I}$ which makes L and all $L \vee F_i$, $\neg L \vee G_j$ and H_k true

❸ $\mathcal{I}$ makes $\neg L$ false, then, since it makes $\neg L \vee G_j$ true, it must make G_j true

❹ $\mathcal{I}$ satisfies $\{G_1, .., \ G_m,\ H_1, .., \ H_p\}$ (by ❸)

❺ contradiction ❷, ❹: S is unsatisfiable

The method of Davis-Putnam

3 Pure-literal rule

If S contains a *pure* literal L, then S' can be obtained by deleting all instances which contain L

- a literal is pure if it only occurs with one sign (positive or negative)

Example

p is pure is S

$$
\begin{array}{rcllcl}
S & = & \{ & p \vee q,\ p \vee \neg q,\ r \vee q,\ r \vee \neg q & \} & \leadsto \quad [\text{rule on } p] \\
 & & \{ & r \vee q,\ r \vee \neg q & \} & \leadsto \quad [\text{rule on } r] \\
S' & = & \{ & & \} & = \quad \emptyset
\end{array}
$$

S' is satisfiable (like S)

The method of Davis-Putnam

Lemma (pure-literal rule)

$S = \{L \vee F_1, .., L \vee F_n, .., G_1, .., G_m\}$ is unsatisfiable iff $\{G_1, .., G_m\}$ is

- *provided L is pure and does not appear in any F_j or G_k*

Proof ($\rightarrow$).

1. S is unsatisfiable
2. suppose $\{G_1, .., G_m\}$ is not: then, there exists $\mathcal{I}$ which makes all G_j true
3. $\mathcal{I}$ can be found which makes L true: therefore, it satisfies all instances $L \vee F_j$, and therefore S
4. contradiction ❷, ❸: $\{G_1, .., G_m\}$ is unsatisfiable

Proof ($\leftarrow$).

easy because $\{G_1, .., G_m\}$ is a subset of the clauses of S

The method of Davis-Putnam

4 Splitting rule

If S takes the form $\{(L \vee F_1), .., (L \vee F_n), (\neg L \vee G_1), .., (\neg L \vee G_m), H_1, .., H_p\}$, then two sets S' and S'' can be obtained as

- $S' = \{F_1, .., F_n, .., H_1, .., H_p\}$
- $S'' = \{G_1, .., G_m, .., H_1, .., H_p\}$

Example

$$S = \{ p \vee \neg q, \neg p \vee q, q \vee \neg r, \neg q \vee \neg r \}$$

$$S' = \{ \neg q, q \vee \neg r, \neg q \vee \neg r \}$$
$$S'' = \{ q, q \vee \neg r, \neg q \vee \neg r \}$$

The method of Davis-Putnam

Lemma (splitting rule)

$S = \{(L \vee F_1), .., (L \vee F_n), (\neg L \vee G_1), .., (\neg L \vee G_m), H_1, .., H_p\}$ is unsatisfiable iff both $S' = \{F_1, .., F_n, .., H_1, .., H_p\}$ and $S'' = \{G_1, .., G_m, .., H_1, .., H_p\}$ are

- *provided neither L nor $\neg L$ appear in any F_i, G_j or H_k*

Proof ($\rightarrow$).

❶ S is unsatisfiable

❷ suppose at least one between S' and S'' is not: therefore, there exists $\mathcal{I}$ which make all H_k true, and either all F_i or all G_j

❸ if $\mathcal{I}$ makes all F_i true, then it makes all $L \vee F_i$ true. $\mathcal{I}$ can be taken which makes L false, so that it makes all $\neg L \vee G_j$ (and S) true

❹ dual reasoning, in the case $\mathcal{I}$ makes all G_j true

❺ in both cases, contradiction (❷, ❸ or ❷, ❹): both S' and S'' are unsatisfiable

The method of Davis-Putnam

Lemma (splitting rule)

$S = \{(L \vee F_1), .., (L \vee F_n), (\neg L \vee G_1), .., (\neg L \vee G_m), H_1, .., H_p\}$ is unsatisfiable iff both $S' = \{F_1, .., F_n, .., H_1, .., H_p\}$ and $S'' = \{G_1, .., G_m, .., H_1, .., H_p\}$ are

- *provided neither L nor $\neg L$ appear in any F_i, G_j or H_k*

Proof ($\leftarrow$).

❶ both S' and S'' are unsatisfiable

❷ suppose S is not: therefore, there exists $\mathcal{I}$ which makes all $L \vee F_i$, $\neg L \vee G_j$ and H_k true

❸ if $\mathcal{I}$ makes L true, then it makes $\neg L$ false: since it makes $\neg L \vee G_j$ true, it must make G_j true, so that it satisfies S''

❹ dual: if $\mathcal{I}$ makes L false, then it satisfies S'

❺ in both cases, contradiction (❷, ❸ or ❷, ❹): S is unsatisfiable

The method of Davis-Putnam

Procedure DP: given S, transform it as follows (YES = satisfiable)

```
while (S ≠ ∅)
   if (tautology rule can be applied) apply tautology rule
   else
      while (one-literal rule can be applied) apply one-literal rule
      if (S contains literals L and ¬L) return NO
      if (S = ∅) return YES
      while (pure-literal rule can be applied) apply pure-literal rule
      if (S contains literals L and ¬L) return NO
      if (S = ∅) return YES
      apply splitting rule, apply DP to both S′ and S″
      if (the result is NO for both S′ and S″) return NO
      else return YES
return YES
```

The Resolution method of Robinson

Our inspiration

In the following part of this section, and the next one, we will (sometimes literally) refer to a couple of papers by John Alan Robinson:

- [R63] Theorem-Proving on the Computer. Journal of the ACM, April 1963, 163-174.
- [R65] A Machine-Oriented Logic Based on the Resolution Principle. Journal of the ACM, January 1965, 23-41.

The Resolution method of Robinson

General idea

Obtaining new instances by deduction from the original set $\mathcal{C}$, such that $\mathcal{C}$ is found to be unsatisfiable whenever both a literal and its negation are deduced

Ground resolution rule

Given two instances $L \vee C_1$ and $\neg L \vee C_2$, where L is a literal, it is possible to deduce a new instance $C_1 \vee C_2$ which is called the *resolvent*

(*Vintage* version of the rule)

- if C and D are two ground clauses, and $L \subseteq C$, $M \subseteq D$ are two singletons (unit sets) whose respective members form a complementary pair, then the ground clause $(C \setminus L) \cup (D \setminus M)$ is called a ground resolvent of C and D [R65]
- if S is any set of ground clauses, then the ground resolution of S, denoted by $\mathcal{R}(S)$, is the set of ground clauses consisting of the members of S together with all ground resolvents of all pairs of members of S [R65]

The Resolution method of Robinson

Unsatisfiability

By applying the rule, it is possible to derive a contradiction when the set is unsatisfiable: such contradiction comes from applying resolution to L and $\neg L$, which generates the *empty clause* $\Box$

Why ground resolution

- as a specific method for testing a finite set of ground clauses for satisfiability, the method of Davis-Putnam would be hard to improve on from the point of view of efficiency [R65]
- now we give another method, far less efficient than theirs, which plays only a theoretical role in our develpment, ... [R65]
- on the other hand, the reason for showing ground resolution is its extension to general resolution

The Resolution method of Robinson

Remark: Idempotence

In order to get a contradition *whenever* the set is unsatisfiable, it is necessary to consider *idempotence* $L \vee L \leftrightarrow L$

Extended resolution

Given two instances $L \vee .. \vee L \vee C_1$ and $\neg L \vee .. \vee \neg L \vee C_2$, it is possible to deduce a resolvent $C_1 \vee C_2$

- Applying this extended rule is called a *resolution step over L with resolvent* $C_1 \vee C_2$

The Resolution method of Robinson

Advantages

The deduction system only consists of one rule

- it is interesting that (as far as the author is aware) no other complete system of first-order logic has consisted of just one inference principle [R65]

Method: given a set S of ground instances

$X = S$
repeat
  generate by resolution steps all possible resolvents from the elements of X:
  let $R(X)$ be the set of resolvents
  if $(\Box \in R(X))$ **then STOP**: $UNSAT(S)$
  if $(R(X) \sqsubseteq X)$ **then STOP**:
    all resolvents have already been generated, so that $SAT(S)$
  $X = R(X) \cup X$

The Resolution method of Robinson

Lemma (Res-1)

Let m be a node of the semantic tree of a set S, and m' and m'' be its direct successors, both failure nodes. The clauses S' and S'' which become false in m' and m'' have a resolvent which is false in m

Proof.

1. m' and m'' are at a level n in the tree, corresponding to the atom A_n; A_n is taken to be true in m' and false in m''
2. $I(m)$ is the partial interpretation in m: $I(m') = I(m) \cup \{A_n\}$ and $I(m'') = I(m) \cup \{\neg A_n\}$
3. S' and S'' take the form, resp., $\neg A_n \vee S'_n$ and $A_n \vee S''_n$, where neither between $\neg A_n$ and A_n appear in S'_n or S''_n
4. $I(m)$ makes both S'_n and S''_n false, since it is not affected by A_n (by ❸)
5. $S'_n \vee S''_n$, which is a resolvent of S' and S'', is false in m (by ❹)

The Resolution method of Robinson

Lemma (Res-2)

Let A be a closed semantic tree where the level of failure nodes is $\leq n$. If m' is a failure node at level n, then its brother m'' is also a failure node

Proof.

1. since the tree is closed, the path through m'' contains a failure node
2. the failure node cannot be after m'', since the maximum level of failure nodes is n, which is the level of m''
3. since m' is a failure node, its predecessors cannot be failure nodes
4. the predecessors of m'' are the same as those of m', so that, by ❸, they cannot be failure nodes
5. by ❶, ❷ and ❹, m'' must be a failure node

The Resolution method of Robinson

Lemma (Res-3)

Let S be an unsatisfiable set of instances which has a closed semantic tree of level n. Then, there exists a set R of resolvents from S such that the semantic tree of $S \cup R$ is closed and has level $n-1$

Proof.

1. every failure node at level n has a brother which is also a failure node (Lemma Res-2)
2. every pair of failure nodes has a resolvent r which is false in their predecessor at level $n-1$ (Lemma Res-1)
3. let $R = \{r \mid r \text{ is the resolvent of two failure nodes at level } n\}$
4. $S \cup R$ has a closed tree of level $n-1$ (by ❸)

The Resolution method of Robinson

Theorem (Res)

A set S of ground instances is unsatisfiable iff it is possible to derive $\Box$ from it by resolution

Proof ($\rightarrow$).

If S is unsatisfiable, then its semantic tree is closed and finite (if pruned at failure nodes). Let n be the maximum level of failure nodes:

- $n = 1$: there are two failure nodes, corresponding to the atom A_1, where A_1 and $\neg A_1$ become false, respectively. The resolvent is $\Box$
- $n > 1$: there exists a set R of resolvents from S such that the semantic tree of $S' = R \cup S$ is closed and has level $n - 1$ (Lemma Res-3)
  - by induction, $\Box$ can be derived from S'
  - however, since S' was derived from S by resolution, $\Box$ can be derived from S

The Resolution method of Robinson

Theorem (Res)

A set S of ground instances is unsatisfiable iff it is possible to derive $\Box$ from it by resolution

Proof ($\longleftarrow$).

1. $S \vdash \Box$ by resolution (where $\Box$ comes as a resolvent of some L and $\neg L$)
2. $S \models \Box$ by ❶ and validity of resolution
3. $\Box$ is false in every interpretation
4. S is false in every interpretation (by ❸ and logical consequence)
5. S is unsatisfiable (by ❹)

The Resolution method of Robinson

General method

- generate all possible sets of ground instances
- for every set, apply ground resolution
- the first step is very inefficient
  - the major combinatorial obstacle to efficiency for level-saturation procedures is the enormous rate of growth of the finite sets H_i and HB_i as i increases [R65]

Example from [R63]

arises from seeking to prove the existence of a right identity element in any algebra closed under a binary associative operation having left and right solutions x and y for all equations $x{\cdot}a = b$ and $a{\cdot}y = b$ whose coefficient a and b are in the algebra

$$\mathcal{C} = \{ \begin{array}{c} \neg p(x,y,u) \vee \neg p(y,z,v) \vee \neg p(x,v,w) \vee p(u,z,w), \\ \neg p(x,y,u) \vee \neg p(y,z,v) \vee \neg p(u,z,w) \vee p(x,v,w), \\ p(g(x,y),x,y), \\ p(x,h(x,y),y), \\ p(x,y,f(x,y)), \\ \neg p(k(x),x,k(x)) \end{array} \}$$

Example from [R63]

- to prove unsatisfiability, only four ground terms (the *proof set* are needed:

$$T = \{\ a,\ h(a,a),\ k(h(a,a)),\ g(a,k(h(a,a)))\ \}$$

- however, in order to get T we need to generate a big (19765) number of terms

Example from [R63]

- moreover, only a negligible part of instances of $\mathcal{C}$ over T is needed to get an unsatisfiable S

$$\begin{array}{lcr}
\{ & p(a,h(a,a),a), & \\
 & \neg p(k(h(a,a)),h(a,a),k(h(a,a))), & \\
 & p(g(a,k(h(a,a))),a,k(h(a,a))), & \\
 & \neg p(g(a,k(h(a,a))),a,k(h(a,a))) \vee \neg p(a,h(a,a),a) \vee & \\
 & \vee \neg p(g(a,k(h(a,a))),a,k(h(a,a))) \vee p(k(h(a,a)),h(a,a),k(h(a,a))) & \}
\end{array}$$

The Resolution method of Robinson

Robinson's idea for efficiency

To postpone the substitution of a variable by a term of the Herbrand universe to when it is needed by some resolution step

- work on clauses with variables
- every resolvent (with variables) represents the set of ground instances which would have been obtained by resolution on ground instances