Propositional logic II: PSAT problems

Readings :
- CHAPTER 7 from Russell + Norvig
- CHAPTERs 13,14 from Nilsson
- Gödel, Escher, Bach.

Some figures from Russell + Norvig

PSAT

- **PSAT = Propositional satisfiability problems**
  Find whether a **wff `w` is satisfiable** (find a model for `w`)
  - » **Soundness**: Correct output (yes if SAT, no if UNSAT)
  - » **Completeness**: Halts and says YES for every SAT wff.
  - » **Decidability**: Halts and says NO for every UNSAT wff.
- Related problems
  - » Find a model for the set of wffs Δ (equivalent to finding a model for the conjunction of wffs in Δ)
  - » Establish whether all the models for the set of wffs Δ are also models for a wff `w` (entailment $\Delta \models w$)

    Consider $\Delta = \{w_1, w_2, ..., w_n\}$

    $\{w_1, w_2, ..., w_n\} \models w$ can be shown by

    – **Deduction:**

    $(w_1 \wedge w_2 \wedge ... \wedge w_n) \Rightarrow w$ is **valid**

    – **Reductio ad absurdum (contradiction)**:

    $(w_1 \wedge w_2 \wedge ... \wedge w_n \wedge \neg w)$ is **unsatisfiable**

- **CNF PSAT problem**: Find a model for a CNF
  - » Solving CNF PSAT is equivalent to solving PSAT
- **kSAT**: Find a model for a conjunction of clauses, the longest of which has k literals.
  - » 2SAT: polynomial complexity
  - » kSAT, $k \geq 3$ is NP-complete.

Does $\Delta \models$ `w` ?

Establish whether all the models for the set of wffs Δ are also models for a wff `w`

- **Model checking** (model enumeration)
  - » Assign all possible combinations of *True* and *False* to the atoms in the formula (computationally expensive: for n atoms 2^n combinations)
  - » Check whether for each assignment where all the fromulas in Δ have value *True*, `w` also has value *True.*

- Solution by **searching for a proof :**
  If we find a set of inference rules *R* sound and complete, we can determine whether w follows from Δ by finding a proof of `w` from Δ

  $\Delta \vdash_R w$

  - » Use a complete search procedure to search on the space of sets of wffs.
  - » The inference rules are the operators that generate successors of the current search state.

Metatheorems, I

- **Metatheorem 3:**

  Deduction theorem

  If $\{w_1, w_2, ..., w_{n-1}, w_n\} \models w$, then

  $(w_1 \wedge w_2 \wedge ... \wedge w_{n-1} \wedge w_n) \Rightarrow w$ is **valid** and viceversa

- **Metatheorem 4:**

  Reductio ad absurdum (contradiction)

  the set of wffs Δ entails w $\Delta \models w$ iff Δ has a model but $\Delta \cup \{\neg w\}$ does not (i.e., there is no interpretation that satisfies all wffs in the combined set).

  $\{w_1, w_2, ..., w_{n-1}, w_n\} \models w$ can be shown by
  - **Deduction**

    $(w_1 \wedge w_2 \wedge ... \wedge w_{n-1} \wedge w_n) \Rightarrow w$ **is a valid wff**

    (i.e., it has the value *True* under all interpretations)
  - **Reductio ad absurdum (Contradiction):**

    $(w_1 \wedge w_2 \wedge ... \wedge w_{n-1} \wedge w_n \wedge \neg w)$ **is an unsatisfiable wff**

    (i.e., there is no interpretation under which it has value *True*)

Metatheorems, II

- **Metatheorem 5:** Let $\lambda_1, \lambda_2, ... \lambda_n$ be n distinct literals. The following statements are equivalent
  - $\bigvee_{i=1}^{n} \lambda_i$ is valid
  - $\bigwedge_{i=1}^{n} \lambda_i$ is unsatisfiable
  - Some pair $(\lambda_i, \lambda_j) 1 \leq i,j \leq n$ is a pair of complementary literals or “mates”$(\lambda_i = \neg\lambda_j)$

Dem.

- It is obvious that if $\lambda_i = \neg\lambda_j$, then $\bigvee_{i=1}^{n} \lambda_i$ is valid and $\bigwedge_{i=1}^{n} \lambda_i$ is unsatisfiable.
- If there is no pair of mates in $\lambda_1, \lambda_2, ... \lambda_n$ then
  - In the interpretation where all $\{\lambda_i\}_{i=1}^{n}$ have value *True,* $\bigwedge_{i=1}^{n} \lambda_i$ has value *True*.
  - In the interpretation where all $\{\lambda_i\}_{i=1}^{n}$ have value *False,* $\bigvee_{i=1}^{n} \lambda_i$ has value *False.*

Metatheorems, III

- **Metatheorem 6:** A formula in CNF is a tautology iff each of its clauses is a tautology.

  Dem.
  - Let α be a wff in CNF
  - Let $K_i,\ i=1,2,...,n$ be the clauses in α

  $$\alpha = K_1 \wedge K_2 \wedge ... \wedge K_n = \bigwedge_{i=1}^{n} K_i$$

  - If each K_i, is a tautology, then, for any interpretation, $K_i\ i=1,2,...,n$ have value *True,* and therefore $\bigwedge_{i=1}^{n} K_i$ has value *True.*
  - If some K_i is not a tautology, then there is at least one interpretation where K_i has value *False*. In that interpretation $\bigwedge_{i=1}^{n} K_i$ has value *False*.

- **Metatheorem 6 (dual):** A formula in DNF is unsatisfiable iff each of its disjuncts is an unsatisfiable conjunction.

Clauses and wffs in CNF as sets

- A clause K is the set $\{\lambda_j\}_{j=1}^{k}$ (also $\bigvee_{j=1}^{k} \lambda_j$) where $\lambda_1,\ \lambda_2, ...\ \lambda_k$ are m different literals
  - Unit clause: A clause containing a single literal $\{\lambda\}$
  - Empty clause (denoted □): A clause with no literals
    - The empty clause is unsatisfiable.

- Let α be a wff in CNF: $\alpha = K_1 \wedge K_2 \wedge ... \wedge K_n = \bigwedge_{i=1}^{n} K_i$ α can be represented by the set of clauses $\{K_i\}_{i=1}^{n}$
  - An empty wff in CNF (empty formula) is a tautology.
  - If K is a tautology, and $K \in \alpha$, then α is satisfiable iff $\alpha - \{K\}$ is satisfiable.

  Important:
  - Order, repetitions are lost in the representation as sets. For truth values, this is irrelevant , because $\vee, \wedge$ are commutative and idempotent.
  - If λ is a negative literal (i.e. $\lambda = \neg P$, P atom), then $\neg\lambda$ is understood to be the equivalent literal P.
  - If we want to prove the satisfiability of α, tautologies can be systematically eliminated, since they are irrelevant in the determination of the satisfiability of α.

Resolution on clauses

Resolution is a rule of inference

- Let λ be an atom and Σ_1, Σ_2 be clauses (sets of literals), not containing either λ or $\neg\lambda$:

  $\Sigma_1\cup\Sigma_2$ can be inferred from $\{\lambda\}\cup\Sigma_1$ and $\{\neg\lambda\}\cup\Sigma_2$

  » **λ is the atom resolved upon**.

  » **$\Sigma_1\cup\Sigma_2$ is a clause**, which is known as **the resolvent** of the two clauses $\{\lambda\}\cup\Sigma_1$ and $\{\neg\lambda\}\cup\Sigma_2$

$$\mathbf{res}_\lambda(\{\lambda\}\cup\Sigma_1,\{\neg\lambda\}\cup\Sigma_2) = \Sigma_1\cup\Sigma_2$$

Particular cases of resolution:

» **Chaining**: From $R\Rightarrow P$ and $P\Rightarrow Q$ infer $R\Rightarrow Q$

Resolving $\neg R\vee P$ and $\neg P\vee Q$ yields $\neg R\vee Q$

» ***Modus ponens***: From R and $R\Rightarrow P$ infer P

Resolving R and $\neg R\vee P$ yields P

» Resolving $P\vee Q\vee R\vee S$ and $\neg P\vee Q\vee W$ yields $Q\vee R\vee S\vee W$

Note: Q does not appear twice (Clauses are sets).

» The wffs $P\vee Q\vee\neg R$ and $P\vee W\vee\neg Q\vee R$ yield

  » Resolving on Q: $P\vee\neg R\vee R\vee W$

  » Resolving on R: $P\vee Q\vee\neg Q\vee W$

Resolution is sound

Soundness:

$\Delta \vdash_R w$ implies $\Delta \models w$

***R** = {resolution on clauses};*

$\Delta = \{\{\lambda\}\cup\Sigma_1 , \{\neg\lambda\}\cup\Sigma_2\}$; $w = \Sigma_1\cup\Sigma_2$

$\Delta\models w$ If the wff w **has value *True*** under **all interpretations** for which **all wffs of Δ** have value ***True***

λ	Σ_1	Σ_2	$\{\lambda\}\cup\Sigma_1$	$\{\neg\lambda\}\cup\Sigma_2$	$\Sigma_1\cup\Sigma_2$
True	*True*	*True*	*True*	*True*	*True*
True	*True*	*False*	*True*	*False*	*True*
True	*False*	*True*	*True*	*True*	*True*
True	*False*	*False*	*True*	*False*	*False*
False	*True*	*True*	*True*	*True*	*True*
False	*True*	*False*	*True*	*True*	*True*
False	*False*	*True*	*False*	*True*	*True*
False	*False*	*False*	*False*	*True*	*False*

Resolution is not complete

Completeness?

Does $\Delta \models w$ imply $\Delta \vdash_R w$?

***R* = {*resolution on clauses*};**

$\Delta = \{\{\lambda\}\cup\Sigma_1 , \{\neg\lambda\}\cup\Sigma_2\}$; $w = \Sigma_1\cup\Sigma_2$

- **Resolution is not complete:**

  Not all logical entailments can be generated by resolution alone.

  Example: The entailment $\{P,Q\} \models P\lor Q$

  cannot be deduced by resolution from the set of clauses $\{P,Q\}$

  But, by resolution refutation:

  $\Delta = \{P,Q,\neg(P\lor Q)\}$ can be shown to be unsatisfiable:

  Using equivalence (De Morgan’s law) $\neg(P\lor Q)\equiv\neg P\land\neg Q$

  $\{P,Q,\neg(P\lor Q)\}\equiv\{P,Q,\neg P,\neg Q\}$

  Using resolution $\{P,Q,\neg P,\neg Q\} \vdash_R \{\Box\}$

Resolution refutation

Given a set of wffs Δ, and a wff w

1. Convert all the wffs in Δ to their equivalent clause form.
2. Convert $\neg w$ to clause form.
3. Combine the clauses resulting from steps 1 and 2 into a single set of clauses α.
4. Perform all possible resolutions in set α, adding the resulting clauses to α.
5. Iterate step (4) until there are no more resolvents than can be added or until the empty clause is produced.

- **Resolution refutation is complete:**

  If, as a result of applying resolution refutation on a set of of wffs Δ, and a wff w the empty clause is produced, then $\Delta \models w$.

  Propositional resolution is **refutation complete**

- **Propositional calculus is decidable by resolution refutation:**

  Let Δ be a set of wffs and a wff w, such that $\Delta \not\models w$, then the resolution refutation procedure terminates without producing the empty clause.

Definitions

- The resolvent of two clauses K_1,K_2 upon the literal λ is $res_\lambda(K_1,K_2)=\{K_1-\{\lambda\}\}\cup\{K_2-\{\neg\lambda\}\}$ $\lambda\in K_1$; $\neg\lambda\in K_2$
- Let λ be a literal, K a clause,
  - K is λ-neutral if $\lambda\notin K, \neg\lambda\notin K$.
  - K is λ-positive if $\lambda\in K$
  - K is λ-negative if $\neg\lambda\in K$

  A clause K is a tautology iff it is both λ-positive and λ-negative for some literal λ.
- **IMPORTANT:** Henceforth **tautologies are removed from all sets of clauses.**

- Let λ be a literal, α a set of clauses,
  - $\alpha = \alpha_\lambda^0 \cup \alpha_\lambda^+ \cup \alpha_\lambda^-$

    $\alpha_\lambda^0= \alpha^0(\lambda)= \{K\in\alpha \,/\lambda\notin K,\neg\lambda\notin K\}$ (λ-neutral K's)

    $\alpha_\lambda^+= \alpha^+(\lambda)= \{K\in\alpha \,/\, \lambda\in K\}$ (λ-positive K's)

    $\alpha_\lambda^-= \alpha^-(\lambda)= \{K\in\alpha \,/\neg\lambda\in K\}$ (λ-negative K's)

    If tautologies are removed, then $\alpha_\lambda^+\cap\alpha_\lambda^-$ is empty.
  - $POS_\lambda(\alpha)= POS(\alpha;\lambda)= \alpha_\lambda^0\cup\{\{K-\{\lambda\}\} \,/\, K\in\alpha_\lambda^+\}$
  - $NEG_\lambda(\alpha)= NEG(\alpha;\lambda)= \alpha_\lambda^0\cup\{\{K-\{\neg\lambda\}\}/\, K\in\alpha_\lambda^-\}$
  - $RES_\lambda(\alpha)= RES(\alpha;\lambda)= \alpha_\lambda^0\cup\{res_\lambda(K_1,K_2)/K_1\in\alpha_\lambda^+,K_2\in\alpha_\lambda^-\}$

Davis-Putnam rules

- **Metatheorem R1 (splitting rule)**: Let λ be a literal. The wff in CNF α is satisfiable iff at least one of the pair $POS_\lambda(\alpha)$, $NEG_\lambda(\alpha)$ is satisfiable.

Demonstration:

- If α is satisfiable, then there is an interpretation under which α has value *True*.
  - If under this interpretation λ has value *True* then $NEG_\lambda(\alpha)=\alpha_\lambda^0\cup\{\{K-\{\neg\lambda\}\}/K\in\alpha_\lambda^-\}$ has value *True*
  - If under this interpretation λ has value *False* then $POS_\lambda(\alpha)=\alpha_\lambda^0\cup\{\{K-\{\lambda\}\}/K\in\alpha_\lambda^-\}$ has value *True*
- If $POS_\lambda(\alpha)$ is satisfiable, construct a model for α from the model of $POS_\lambda(\alpha)$ + the assignment *True* for λ.
- If $NEG_\lambda(\alpha)$ is satisfiable, construct a model for α from the model of $NEG_\lambda(\alpha)$ + the assignment *False* for λ.
  - Corollary I (**pure literal rule**): If α_λ^- contains no clauses (i.e., it is the empty formula), then α is satisfiable iff $NEG_\lambda(\alpha)= \alpha_\lambda^0$ is satisfiable.
  - Corollary II (**unit clause rule**): If the unit clause $\{\lambda\}\in\alpha$ then α is satisfiable iff $NEG_\lambda(\alpha)$ is satisfiable.

Resolution theorem

- **Metatheorem R2**: Let λ be a literal. The wff in CNF α is satisfiable iff $\mathrm{RES}_\lambda(\alpha)$ is satisfiable.

Demonstration:

» If α is satisfiable, then there is an interpretation under which α has value *True*.

For $K \in \mathrm{RES}_\lambda(\alpha)$

  - » $K \in \alpha_\lambda^0$ then, $K \in \alpha$, and, under this interpretation, K has value *True.*
  - » $K = res_\lambda(K_1, K_2)$ for some $K_1 \in \alpha_\lambda^+, K_2 \in \alpha_\lambda^-$. Since K_1, K_2 have value *True* under this interpretation, K also has value *True.*

Hence, $\mathrm{RES}_\lambda(\alpha)$ is satisfiable.

» If $\mathrm{RES}_\lambda(\alpha)$ is satisfiable, then there is an interpretation under which $\mathrm{RES}_\lambda(\alpha)$ has value *True*.

Under this interpretation α_λ^0 has value *True.*

Under this interpretation, either $\mathrm{POS}_\lambda(\alpha)$ or $\mathrm{NEG}_\lambda(\alpha)$ has value *True*.

  - » Assume $\mathrm{POS}_\lambda(\alpha)$ has value *False.* Then there must be a $K_1 \in \alpha_\lambda^+$ for which $\{K_1 - \{\lambda\}\}$ has value *False.*
  - » Since all resovents $\{res_\lambda(K_1, K_2),\ K_2 \in \alpha_\lambda^-\}$ have value *True*, then all $K_2 \in \alpha_\lambda^-$ $\{K_2 - \{\neg\lambda\}\}$ have value *True*. This implies that $\mathrm{NEG}_\lambda(\alpha)$ has value *True.*

Since at least one of the pair $\mathrm{POS}_\lambda(\alpha), \mathrm{NEG}_\lambda(\alpha)$ is satisfiable, by metatheorem 1, α is satisfiable.

Resolution derivation

- **Resolution derivation**: Let Δ be a finite set of clauses.

  A sequence of clauses $K_1, K_2, ..., K_n = K$ is called a resolution derivation of K from Δ if either $K \in \Delta$ or for each K_i, $i = 1, 2, ..., n$, there are K_j, K_k, $j, k < i$ and a literal λ such that $K_i = res_\lambda(K_j, K_k)$.

- **Metatheorem R3**: Assume a resolution derivation of K from the wff in CNF Δ exists. Then $\Delta \models K$.

  Demonstration, by induction:

  - » Let $K_1, K_2, ..., K_n = K$ be a resolution derivation of K from Δ.
  - » Consider an interpretation for which Δ has value *True*.
    - » if $K \in \Delta$ then K has, under this interpretation, the value *True.*
    - » Assume $K_1, K_2, ..., K_{i-1}$ all have value *True* under this interpretation, then
      - » If $K_i \in \Delta$ then K_i has the value *True*
      - » If $K_i = res_\lambda(K_j, K_k)$ $j, k < i$ then K_i also has the value *True*

        (because K_j, K_k, $j, k < i$, have value *True* and resolution is sound)

Ground Resolution theorem

- **Resolution refutation**: A **resolution derivation** of the **empty clause from** α is a **resolution refutation of** α.
- **<u>Metatheorem R4</u>**: The wff in CNF α is unsatisfiable iff there is a resolution refutation of α.

  <u>Demonstration:</u>
  - » Assume there is a resolution refutation of α.
    - (1) Assume there is an interpretation under which α has value *True.*
    - (2) From metatheorem 3, it follows that the empty clause `{}` would have value *True* under that interpretation.
    - (3) Since `{}` is unsatisfiable, (1) cannot obtain.

    There cannot be any interpretation under which α has value *True.*
  - » Assume α is unsatisfiable.
    - (1) $\lambda_1, \lambda_2, ... \lambda_k$ is the list of different atoms in α.
    - (2) Let $\alpha_0 = \alpha$, $\alpha_i = \text{RES}(\alpha_{i-1};\lambda_i)$, $i=1,2,...,k$

      By metatheorem 2, all α_i are unsatisfiable

      α_k is unsatisfiable and contains no atoms. Therefore, it must contain only the empty clause.
    - (3) Let $K_0, K_1, K_2, ..., K_m$ be the sequence of clauses, first of α_0, then of α_1 and so on, up to the only clause of α_k. K_m is the empty clause.

      $K_0, K_1, ..., K_m$ is a resolution refutation of α.

Deletion rules that can be used in resolution strategies

- **Remove repeated literals in clauses.**
- **Remove repeated clauses.**
- **Remove clauses that are tautologies**
- **Remove subsumed clauses.**
  - » If $K_1 \subset K_2$, then we say that

    K_2 is **subsumed** by K_1 (K_1 subsumes K_2)
  - » If $K_1 \subset K_2$, then $K_1 \models K_2$
  - » If both $K_1, K_2 \in \alpha$ and K_2 is subsumed by K_1 ($K_1 \subset K_2$) then α is satisfiable iff $\alpha - \{K_2\}$satisfiable.

  <u>Example</u>:

  $K_1 = P \vee Q \vee R$

  $K_2 = P \vee Q \vee R \vee S$

  $K_1 \subset K_2$ (K_1 subsumes K_2)

  If both are present in α, then remove K_2 from α

- **Apply Davis-Putnam rules**
  - » **Pure literal rule:** Check the satisfiability only of those clauses that do not contain the pure literals
  - » **Unit clause rule:**Check the satisfiability of $\text{NEG}_\lambda(\alpha)$ for those literals λ that appear in unit clauses.

Example: Use of Davis Putnam rules

Is α satisfiable?

$\alpha = \{\neg P\vee\neg Q\vee\neg R, \neg P\vee S, P\vee S, Q\vee R, \neg Q\vee S, \neg Q\vee P, P, Q, \neg R, \neg P\vee\neg Q\vee R\}$

- Use **pure literal rule**: Check the satisfiability only of those clauses that do not contain the pure literals

  S is a pure literal

  $\alpha_1 = \{\neg P\vee\neg Q\vee\neg R, Q\vee R, \neg Q\vee P, P, Q, \neg R, \neg P\vee\neg Q\vee R\}$

- Use **unit clause rule:** Check the satisfiability of $NEG_{\lambda}(\alpha)$ for those literals λ that appear in unit clauses.

  $\alpha_2 = NEG_Q(\alpha_1) = \{\neg P\vee\neg R, P, \neg R, \neg P\vee R\}$

  $\alpha_3 = NEG_{\neg R}(\alpha_2) = POS_R(\alpha_2) = \{P, \neg P\}$

  $\alpha_4 = NEG_P(\alpha_2) = \{\square\}$

α is not satisfiable

Resolution closure

The **resolution closure** of a **set of clauses** α, **RC(α),** is the set of all **clauses** that can be **deduced** by **repeated application of resolution**.

Let $\lambda_1, \lambda_2, ...\lambda_{k-1}, \lambda_k$ are k different atoms that make up the clauses of α.

Let $\alpha_0 = \alpha$,

$\alpha_j = RES(\alpha_{j-1}; \lambda_j),\ j=1,2,...,k$

Then, by construction $RC(\alpha) = \alpha_k$.

Properties:

- **RC(α)** is **finite** (α contains a finite set of atoms)
- **RC(α) is closed under resolution**.

From the resolution metatheorems:

- **RC(α)** contains the empty clause iff the set of clauses α is unsatisfiable.
- **RC(α)** does not contain the empty clause iff the set of clauses α is satisfiable.

A direct resolution strategy (I)

Is set of clauses α SAT?

Let $\lambda_1, \lambda_2, ...\lambda_{k-1}, \lambda_k$ are k different atoms that make up the clauses of α.

Let $\alpha_0=\alpha$,

$\alpha_j=\mathrm{RES}(\alpha_{j-1};\lambda_j)$, (tautologies removed from α_j)

$j=1,2...,k$

until no more resolutions can be made or
until the empty clause is produced.

Example:

$\alpha = \{P\vee Q, \neg P\vee Q, P\vee\neg Q\}$

The list of different atoms of α is P,Q.

- » $\alpha_0 = \alpha = \{P\vee Q, \neg P\vee Q, P\vee\neg Q\}$
  - – $\alpha^0(P) = \{\}$
  - – $\alpha^+(P) = \{P\vee Q, P\vee\neg Q\}$
  - – $\alpha^-(P) = \{\neg P\vee Q\}$
- » $\alpha_1 = \mathrm{RES}(\alpha_0;P) = \{Q, Q\vee\neg Q\} \equiv \{Q\}$
  - – $\alpha_1^0(Q) = \{\}$
  - – $\alpha_1^+(Q) = \{Q\}$
  - – $\alpha_1^-(Q) = \{\}$
- » $\alpha_2 = \mathrm{RES}(\alpha_1;Q) = \{\}$ (empty set of clauses)

Therefore, α is SAT

A direct resolution strategy (II)

Example:

$\alpha = \{P\vee Q, \neg P\vee Q, P\vee\neg Q, \neg P\vee\neg Q\}$

The list of different atoms of α is P,Q.

- » $\alpha_0 = \alpha = \{P\vee Q, \neg P\vee Q, P\vee\neg Q, \neg P\vee\neg Q\}$
  - – $\alpha^0(P) = \{\}$
  - – $\alpha^+(P) = \{P\vee Q, P\vee\neg Q\}$
  - – $\alpha^-(P) = \{\neg P\vee Q\ \neg P\vee\neg Q\}$
- » $\alpha_1 = \mathrm{RES}(\alpha_0;P) = \{Q, Q\vee\neg Q, \neg Q\} \equiv \{Q, \neg Q\}$
  - – $\alpha_1^0(Q) = \{\}$
  - – $\alpha_1^+(Q) = \{Q\}$
  - – $\alpha_1^-(Q) = \{\neg Q\}$
- » $\alpha_2 = \mathrm{RES}(\alpha_1;Q) = \{\Box\}$

  (a set with the empty clause)

Therefore, α is UNSAT

Example

Does $\Delta \models w$?

$\Delta = \{P \Leftrightarrow (Q \vee R), \neg P\}$

$w = \neg Q$

- Convert to CNF

  $\Delta \equiv \{\neg P \vee Q \vee R, \neg Q \vee P, \neg R \vee P, \neg P\}$

  $\neg w \equiv Q$
- Construct α

  $\alpha = \{\neg P \vee Q \vee R, \neg Q \vee P, \neg R \vee P, \neg P, Q\}$
- Use metatheorems R2 and R4, eliminating tautologies
  The list of different atoms of α is P,Q,R.
  - $\alpha_0 = \alpha = \{\neg P \vee Q \vee R, \neg Q \vee P, \neg R \vee P, \neg P, Q\}$
    - $\alpha^0(P) = \{Q\}$
    - $\alpha^+(P) = \{\neg Q \vee P, \neg R \vee P\}$
    - $\alpha^-(P) = \{\neg P \vee Q \vee R, \neg P\}$
  - $\alpha_1 = RES(\alpha_0;P) =$
    $= \{Q, \neg Q \vee Q \vee R, \neg Q, \neg R \vee Q \vee R, \neg R\} \equiv \{Q, \neg Q, \neg R\}$
    - $\alpha^0(Q) = \{\neg R\}$
    - $\alpha^+(Q) = \{Q\}$
    - $\alpha^-(Q) = \{\neg Q\}$
  - $\alpha_2 = RES(\alpha_1;Q) = \{\neg R, \square\}$

  α is not satisfiable

Level saturation resolution

- **Level saturation resolution:**
  - $\alpha_0 = \alpha$
  - $\alpha_i = \{$all possible resolvents between α_{i-1} and $\alpha_1 \cup \alpha_2 \cup \ldots \cup \alpha_{i-1}\}$, $i=1,2,3,\ldots$

    until no more resolvents than can be added or until the empty clause is produced.

- **Example:**

  $\alpha = \{P \vee Q, \neg P \vee Q, P \vee \neg Q, \neg P \vee \neg Q\}$

  Note: both α and the clauses in α are defined as sets (i.e. Order is not important. Repeated clauses in α can be listed only once. Repeated literals in a clause can be listed as a single literal). Clauses that are tautologies can be removed from the KB.
  - $\alpha_0 = \alpha = \{P \vee Q, \neg P \vee Q, P \vee \neg Q, \neg P \vee \neg Q\}$
  - $\alpha_1 = \{Q \vee Q, P \vee P, Q \vee \neg Q, P \vee \neg P, Q \vee \neg Q,$
    $P \vee \neg P, \neg P \vee \neg P, \neg Q \vee \neg Q\} \equiv \{Q, P, \neg P, \neg Q\}$

    $\alpha_0 \cup \alpha_1 = \{P, Q, \neg P, \neg Q, P \vee Q, \neg P \vee Q, P \vee \neg Q, \neg P \vee \neg Q\}$
  - $\alpha_2 = \{\square, P, Q, \neg P, \neg Q, P \vee Q, \neg P \vee Q, P \vee \neg Q, \neg P \vee \neg Q\}$

Unit resolution

Restricted form of resolution where every resolution step involves a unit clause (clause with a single literal)

- » **Efficient.**
- » **Not complete** in general.
- » Complete for Horn knowledge bases.

Example: α = {P∨Q,¬P∨R ,¬Q∨R,¬R}

```
(1) P∨Q    (2) ¬P∨R (3) ¬Q∨R (4) ¬R
-----------
(5) {¬P} [(2)+(4)]
(6) {¬Q} [(3)+(4)]
-----------
(7) {Q} [(1)+(5)]
(8) {P} [(1)+(6)]
-----------
(10) {R} [(3)+(7)]
(11) {□} [(6)+(7)]
(12) {R} [(2)+(8)]
(13) {□} [(5)+(8)]
```

- **Unit preference resolution:** Prefer resolutions with unit clauses

  Why?: Since we are trying to generate the empty clause, prefer inferences that lead to shorter clauses.

  - » Introduced first in 1964.
  - » Speedup not sufficient to solve medium-sized problems.

Set of support resolution

- The **set of support** of a set of clauses α is the subset Γ such that the set α-Γ is satisfiable.

- **Set of support resolution**

  Restricted form of resolution in which every resolution step must involve at least a clause from the set of support.

  The resolvent clause is then included in Γ.

  - » **Complete if α-Γ is satisfiable**. Not complete for a wrong choice for the initial set of support.
  - » In entailment problems (Δ⊨w ?, α = {Δ∧¬w}), a common choice is to initialize Γ with the result of negating the initial query (Γ = {¬w})
    - – This choice is correct if the original knowledge base Δ is consistent (satisfiable). Note that if the original KB is not consistent, any statement can be derived from it.
    - – It is a goal-oriented refutation (intelligible!)

Example: Δ = {P∨Q,¬P∨R ,¬Q∨R}; w = R.

α = {P∨Q,¬P∨R ,¬Q∨R,¬R}

(0) Γ_0 = {¬R};
(1) Γ_1 = {¬R,¬P};
(2) Γ_2 = {¬R,¬P,¬Q};
(3) Γ_3 = {¬R,¬P,¬Q,Q};
(4) Γ_4 = {¬R,¬P,¬Q,Q,P,□};

Input and linear resolution

- **Input resolution**

  Resolution in which every resolution step involves a clause from the initial set of clauses and some other clause

  - » **Not complete** in general.
  - » Complete for knowledge bases in the Horn form.

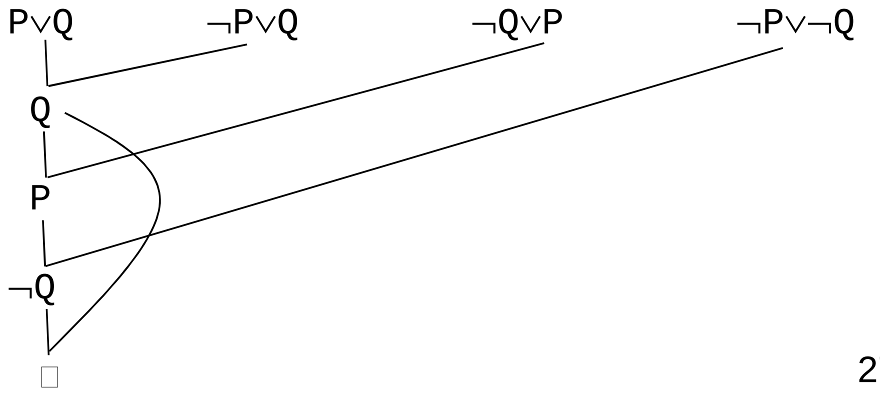
- **Linear resolution (ancestry-filtered resolution)**

  Generalization of input resolution in which every resolution step must involve some clause and a clause from either the initial set of clauses or from an antecessor of the current clause in the derivation tree

  - » **Complete.**

Example: $\alpha = \{P\vee Q, \neg P\vee Q, \neg Q\vee P, \neg P\vee\neg Q\}$

Input resolution fails



Horn clauses

A **Horn clause** is a non-empty set of literals of which **at most one is a positive literal**.

- A **Horn clause** is either
  - **A "consistency constraint":** A clause with **no positive literals** (e.g., the negation of a conjunctive goal clause)

    E.g. $\neg P\vee\neg Q \equiv \neg(P\wedge Q) \equiv (P\wedge Q)\Rightarrow F$
  - a **"fact"**: a unit clause with a single positive literal

    E.g. P,Q (facts)
  - **A "rule":** a clause with a single positive literal and with at least one negative literal. A rule can be written as an equivalent **implication**, where the premise is the conjunction of the negations of the negative literals, and the consequent is the positive literal.

    E.g. $\neg P\vee\neg Q\vee\neg R\vee S \equiv (P\wedge Q\wedge R)\Rightarrow S$ (rule)
- Inference on a set of Horn clauses α can be made by
  - » **Forward chaining** (data-driven reasoning)
  - » **Backward chaining** (goal-directed reasoning)

» **SAT on** a set of **Horn clauses is linear** in the number of clauses in the set.

Forward chaining

- **Consider a set of Horn clauses α_0**

  1. $\alpha = \alpha_0$

  2. Apply those rules whose premise are in α, and incorporate the conclusions into α.

  3. Repeat (2) until no more rules can be applied.
- Forward Chaining (FC) on sets of Horn clauses
  - **FC is sound:** Each inference is an application of modus ponens, which is sound.
  - **FC is complete:** FC derives every atomic sentence that is entailed by α_0

  Demonstration:
  - When FC terminates no new atomic sentences can be deduced by application of FC.
  - Interpretation where all the atoms deduced by FC from α_0 are assigned value *True* and the atoms not deduced by FC from α_0 are assigned value *False.*
  - This interpretation is a model of α_0 .

    By contradiction: Assume the rule $(P_1 \wedge P_2 \wedge ... \wedge P_n) \Rightarrow S$ in α_0 has value *False*. $(P_1 \wedge P_2 \wedge ... \wedge P_n)$ must have value *True* and S must have value *False.* Then $P_1, P_2, ..., P_n$ have value *True* i.e. they have been deduced by FC, and the rule $(P_1 \wedge P_2 \wedge ... \wedge P_n) \Rightarrow S$ should have been applied.
  - Assume there is an atom $P / \alpha_0 \models P$ then P has value True in all models of α_0, and P must be one of the atoms deduced by FC from α_0.

Backward chaining

- **Deduce goal P from a set of Horn clauses α**

  1. If $P \in \alpha$ then goal found.

  2. Prove by backward chaining all atoms in the premise of some rule in α whose conclusion is P.

- Avoid loops:

  Check whether new subgoal is already on goal stack.

- Check whether it has already been shown that the new subgoal
  - can be deduced from α.
  - cannot be deduced from α.

- Backward vs Forward chaining
  - **FC is data-driven**, appropriate for acquisition of knowledge without a specific query in mind.
  - **BC is goal-directed**, appropriate for solving specific problems. The complexity of BC can be much less than linear in size of KB.

An agent should use FC to generate facts that are likely to be relevant to queries solved by BC.

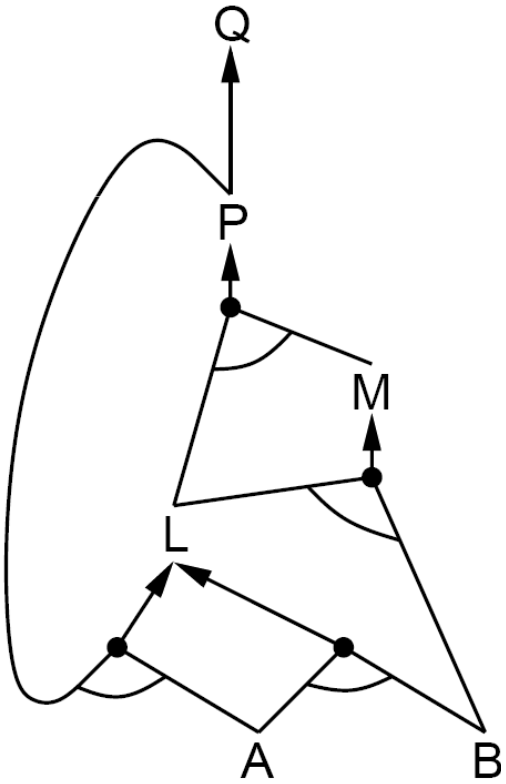
Forward/Backward chaining: AND-OR diagram

- Example (from Russel + Norvig)

  Consider an agent whose knowledge base is

  Facts: A, B

  Rules: $P \Rightarrow Q$, $L \wedge M \Rightarrow P$, $B \wedge L \Rightarrow M$, $A \wedge P \Rightarrow L$, $A \wedge B \Rightarrow L$



Entailment by Forward/Backward chaining

Consider

$\Delta = \{ P, N, P \Rightarrow Q, Q \Rightarrow R, R \wedge S \Rightarrow M, N \Rightarrow S\}$

$w = M \wedge R$

Does $\Delta \models w$?

- » Construct $\alpha = \Delta \cup \neg w$
- » Note that $\neg w$ is a Horn clause
  $\neg(M \wedge R) \equiv \neg M \vee \neg R \equiv M \wedge R \Rightarrow F$
- » Show that α is unsatisfiable either by
  - – Forward chaining:

    Facts: $\{P,N\}$

    $\{P,N,Q,S\}$ [by rules $P \Rightarrow Q, N \Rightarrow S$]

    $\{P,N,Q,S,R\}$ [by rule $Q \Rightarrow R$]

    $\{P,N,Q,S,R,M\}$ [by rule $R \wedge S \Rightarrow M$]

    $\{P,N,Q,S,R,M,F\}$ [by rule $M \wedge R \Rightarrow F$]
  - – Backward chaining:

    F can be inferred from M[1],R[2]

    (1) M can be inferred from R[1.1]=[2],S[1.2]

    (1.2) S can be inferred from N[1.2.1]

    (1.2.1) N is in knowledge base.

    (2) R can be inferred from Q [2.1]

    (2.1) Q can be inferred from P [2.1.1]

    (2.1.1) P is in knowledge base.

Model checking for $\Delta \models$ `w`

Establish whether all the models for the set of wffs Δ are also models for a wff `w`

- **Model checking (model enumeration):**
  - Assign all possible combinations of *True* and *False* to the atoms in the formula.
  - Check whether for each assignment where all the fromulas in Δ have value *True*, `w` also has value *True.*
- This procedure is
  - Sound: Direct application of the definition of entailment.
  - Complete: Based on an exhaustive search in the space of interpretations (which is finite)
  - Time Complexity: O(2^n ; n = number of atoms)
  - Space complexity O(n)

Algorithms:

- **DPLL algorithm** (Davis, Putnam, Logemann, Loveland).
- **GSAT.**
- **WALKSAT** (random walk variation of GSAT)

The DPLL algorithm

- **Davis, Putnam, Logemann, Loveland (1962)**
  - Is the set of clauses α satisfiable?
  - Recursive, depth first enumeration of possible models.
  - **Early termination**:
    - A clause has value *True* if any of its literals has value *True.*
    - A set of clauses has value *False* if any of its clauses has value *False.*
  - Use **Davis-Putnam rules**
    - **Pure literal rule**:
      - If α_λ^- is empty and α_λ^+ is not empty, a model for α must assign λ the value *True.*
      - If α_λ^+ is empty and α_λ^- is not empty, a model for α must assign the value *False to* λ.
    - **Unit clause rule**:
      - If the unit clause $\{\lambda\} \in \alpha$ then a model for α must assign λ the value *True.*
      - If in any clause in α all literals except for λ, have been assigned *False*, then a model for α must assign λ the value *True.*

Local search algorithms for SAT

Is the set of clauses α satisfiable?

- GSAT

  1. Select a random set of truth values for all atoms in α.

  2. Use greedy local search to maximize the number of clauses that have value *True* by modifying the value of one single atom at a time.

  » Not optimal (may get trapped in local maxima).

  `Example:  α ={¬P∨¬Q∨¬R,¬P∨S,P∨S,Q∨R,¬Q∨S,¬Q∨P`
  `P,Q,¬R,¬P∨¬Q∨R}`

P	Q	R	S	α	Number of clauses satisfied
True	***False***	***True***	***False***	***False***	***7***
False	*False*	*True*	*False*	*False*	*6*
True	*True*	*True*	*False*	*False*	*6*
True	*False*	*False*	*False*	*False*	*7*
True	***False***	***True***	***True***	***False***	***8***
…					

- WALKSAT

  Occasionally perform a random walk step instead of a steepest ascent step.

  » If α is unsatisfiable, WALKSAT with unlimited steps never terminates.

Hard SAT problems

Considering random sets of 3-CNF wffs with
m clauses
n different symbols,

Hard problems for WALKSAT and DPLL tend to cluster around a “critical point” m/n = 4.3