1 Explanation Generation

Algorithm 1: ExplainCSP($\mathcal{C}, U, f, I$)

input : $\mathcal{C}$ *a CNF* $\mathcal{C}$ *over a vocabulary* V
input : U *a user vocabulary* $U \subseteq V$
input : f*, a cost function* $f : 2^{\mathcal{G}} \rightarrow \mathbb{N}$ *over a CNF* $\mathcal{G}$
input : I*, a partial interpretation over* U
output : E*, a sequence of explanation steps as implications* $I_{expl} \implies N_{expl}$

$E \leftarrow \langle\rangle$
$I_{end} \leftarrow$ OptimalPropagate($\mathcal{C} \cup I, U$) `// assignment on variables of U`
while $I \neq I_{end}$ **do**
  $X \leftarrow$ bestStep($\mathcal{C}, f, I_{end}, I$)
  $I_{best} \leftarrow I \cap X$
  $N_{best} \leftarrow$ OptimalPropagate($C \cup I_{best}, U$) $\setminus I$
  add $\{I_{best} \implies N_{best}\}$ to E
  $I \leftarrow I \cup N_{best}$
end
return E

Algorithm 2: OptimalPropagate(sat,$\mathcal{U}[, I]$)

input : sat*, a* sat *solver bootstrapped with a CNF.*
input : U *a user vocabulary* $U \subseteq V$
optional: I*, a set of assumption literals.*
output : *The projection onto* U *of the intersection of all models of* U

$sat?, \mu \leftarrow$ sat(I)
$\mu \leftarrow \{x \mid x \in \mu : \texttt{var}(x) \in U\}$
$b_i \leftarrow$ a new blocking variable
while *true* **do**
  $\mathcal{C} \leftarrow \mathcal{C} \wedge (\neg b_i \bigvee_{x \in \mu} \neg x)$
  $sat?, \mu' \leftarrow$ sat($I \wedge \{b_i\}$)
  if $\neg sat?$ **then**
    add clause ($\neg b_i$) to sat solver
    return μ
  end
  $\mu \leftarrow \mu \cap \{x' \mid x' \in \mu' : \texttt{var}(x') \in U\}$
end

Algorithm 3: bestStep–c-OUS($\mathcal{C}, f, I_{end}, I$)

input : $\mathcal{C}$*, a CNF.*
input : f*, a cost function* $f : 2^{\mathcal{G}} \rightarrow \mathbb{N}$ *over CNF* $\mathcal{G}$*.*
input : I_{end}*, the cautious consequence, the set of literals that hold in all models.*
input : I*, a partial interpretation s.t.* $I \subseteq I_{end}$*.*
output : *a single best explanation step*

$A \leftarrow I \cup (\overline{I_{end}} \setminus \bar{I})$ `// Optimal US is subset of A`
set $p \triangleq \sum_{l \in \overline{I_{end}}} l = 1$ i.e. exactly one of $\overline{I_{end}}$ is present in the hitting set
return c-OUS($\mathcal{C}, f, p, A$)

Algorithm 4: c-OUS$(\mathcal{C}, f, p, A)$

input : $\mathcal{C}$, *a CNF.*
input : f, *a cost function* $f : 2^{\mathcal{G}} \rightarrow \mathbb{N}$ *over CNF* $\mathcal{G}$.
input : p, *a predicate* $p : 2^{\mathcal{G}} \rightarrow \{t, f\}$ *over CNF* $\mathcal{G}$.
input : A, *a set of assumption literals, s.t.* $\mathcal{C} \cup A$ *is unsatisfiable.*
output : *a* p*-constrained* f*-optimal unsatisfiable subset* $(p, f) - OUS$.

$\mathcal{H} \leftarrow \emptyset$
while *true* **do**
  $A' \leftarrow$ CondOptHittingSet$(f, p, A, \mathcal{H})$
  if $\neg$sat$(\mathcal{C} \cup A')$ **then**
    return A'
  end
  ~~$A'' \leftarrow$ Grow(C, f, p, A', A)~~
  Optional Grow, if the sat solver can provide a provide a good model, we can skip the expensive call to the grow procedure. Needs to be checked experimentally!
  ~~$\mathcal{H} \leftarrow \mathcal{H} \cup \{A \setminus A''\}$~~
  $\mathcal{H} \leftarrow \mathcal{H} \cup \{A \setminus A'\}$
end