1 Explanation Generation

Algorithm 1: CSP-Explain($\mathcal{T}$, f [, $\mathcal{I}_0$])

input : $\mathcal{T}$ *set of constraints*
input : f *a consistent objective function*
optional input: $\mathcal{I}_0$ *a partial interpretation*
output : *Explanation sequence*

begin
  $\mathcal{I} \leftarrow \mathcal{I}_0$ `// Initial partial interpretation`
  $\mathcal{I}_{end} \leftarrow$ `propagate(`$\mathcal{I}$, $\mathcal{T}$`)` `// Goal state`
  $Seq \leftarrow$ *empty set* `// explanation sequence`
  while $\mathcal{I} \neq \mathcal{I}_{end}$ **do**
    $\mathcal{F} \leftarrow \mathcal{I}_{end} \setminus \mathcal{I}$; `// Facts to be derived`
    $\mathcal{F}' \leftarrow \{\neg\mathcal{F}\}$; `// Set with all negated literals of` $\mathcal{F}$
    `//`
    `// Propagate all negated literals OMUS finds the smallest explanation`
    `//`
    $X \leftarrow$ `OMUS(`$\mathcal{F}' \wedge \mathcal{I} \wedge \mathcal{S}$`)` ;
    $E \leftarrow \mathcal{I} \cap X$; `// Explanation used`
    $\mathcal{N} \leftarrow$ `propagate(`$E \wedge \mathcal{S}$`)` ; `// Newly derived facts`
    $\mathcal{I} \leftarrow \mathcal{I} \cup \mathcal{N}$; `// Update known facts`
    `//`
    `// Add explanation for newly derived facts to explanation sequence`
    `//`
    for $n \in \mathcal{N}$ **do**
      (E_n, $\mathcal{S}_n$, n) to Seq ;
    end
  end
end

2 OMUS Algorithm

Note that if we assign a unit weight to every element in the subset, we reduce the problem of finding an OMUS to finding a SMUS.

Definition 1. *Let Γ be a collection of sets and HS(Γ) the set of all hitting sets on Γ and let f be an valid objective function. Then a hitting set $h \in \Gamma$ is said to be an **optimal** hitting set if $\forall\ h' \in HS(\Gamma)$ we have that $f(h) \leq f(h')$.*

Property 1. *The **optimal** hitting set of a collection of sets Γ is denoted by $OHS(\Gamma)$.*

The algorithm is based on the following observation:

Proposition 1. *A set $\mathcal{U} \subseteq \mathcal{F}$ is an OMUS of $\mathcal{F}$ if and only if $\mathcal{U}$ is an optimal hitting set of MCSes($\mathcal{F}$)*

Lemma 1. *Let $\mathcal{K} \subseteq$ MCSes($\mathcal{F}$). Then a subset $\mathcal{U}$ of $\mathcal{F}$ is an OMUS if $\mathcal{U}$ is a optimal hitting set on $\mathcal{K}$ and $\mathcal{U}$ is unsatisfiable*

Algorithm 2: OMUS-Delayed($\mathcal{F}$, $[f\ , \mathcal{H}_0]$)

input : *$\mathcal{F}$ a CNF formula*
input : *cost a cost function*
optional input: *$\mathcal{H}_0$ initial collection of disjoint Minimum Correction Sets*
output : $\mathcal{OMUS}(\mathcal{F})$

begin
    $\mathcal{H} \leftarrow$ `DisjointMCS`$(\mathcal{F})$
    while *true* **do**
        $hs \leftarrow$ `OptimalHittingSet`$(\mathcal{H}, cost)$ `// Find optimal solution`
        $\mathcal{F}' \leftarrow \{c_i | e_i \in hs\}$
        (sat?, κ) $\leftarrow$ `SatSolver`$(\mathcal{F}')$
        `// If SAT,` κ `contains the satisfying truth assignment`
        `// IF UNSAT,` $\mathcal{F}'$ `is the OMUS`
        if *not sat?* **then**
            break
            $\mathcal{C} \leftarrow \mathcal{F} \setminus$ `Grow`$(\mathcal{F}')$
            $\mathcal{H} \leftarrow \mathcal{H} \cup \{\mathcal{C}\}$
            $nonOptLevel \leftarrow 0$
            `// Find a series of non-optimal solutions`
            while *true* **do**
                switch *nonOptLevel* **do**
                    case *0*
                        `// adds clause to current core to the current hitting set`
                        $hs \leftarrow$ `FindIncrementalHittingSet`$(H, \mathcal{F}', hs)$
                    case *1*
                        `// greedy algorithm`
                        $hs \leftarrow$ `FindGreedyHittingSet`$(\mathcal{H})$
                end
                $\mathcal{F}' \leftarrow \{c_i | e_i \in hs\}$
                (sat?, κ) $\leftarrow$ `SatSolver`$(\mathcal{F}')$
                if *not sat?* **then**
                    switch *nonOptLevel* **do**
                        case *0*
                            $nonOptLevel \leftarrow 1$
                        case *1*
                            break
                    end
                else
                    $\mathcal{C} \leftarrow \mathcal{F} \setminus$ `Grow`$(\mathcal{F}')$
                    $\mathcal{H} \leftarrow \mathcal{H} \cup \{\mathcal{C}\}$
                    $nonOptLevel \leftarrow 0$
            end
        end
    end
    return $(\mathcal{F}',\ cost(\mathcal{F}'))$
end

3 MIP hitting set problem specification

For the set of clauses $\mathcal{C} = \{c_1, ...c_{|C|}\}$ with weights $\mathcal{W} = \{w_1, ...w_{|C|}\}$ in the collection of sets $\mathcal{H}$. For Example:

$$\begin{aligned} \mathcal{C} &= \{c_1, ...c_6\} \\ \mathcal{W} &= \{w_1 = 20, w_2 = 20, w_3 = 10, w_4 = 10, w_5 = 10, w_6 = 20\} \\ \mathcal{H} &= \{c_3\},\ \{c_2, c_4\},\ \{c_1, c_4\},\ \{c_1, c_5, c_6\} \end{aligned} \tag{1}$$

The optimal hitting set can be formulated as an integer linear program.

$$min \sum_{i \in \{1..|C|\}} w_i \cdot x_i \tag{2}$$

$$\sum_{i \in \{1..|C|\}} x_i \cdot h_{ij} \geq 1,\ \forall\ j \in \{1..|\mathcal{H}|\} \tag{3}$$

$$x_i = \{0, 1\} \tag{4}$$

- w_i is the input cost/weight associated with clause i in
- x_i is a boolean decision variable if constraint/clause c_i is chosen or not.
- Equation 3, h_{ij} is a boolean input variable corresponding to if constraint/clause i is in set to hit j.

4 Future Work

References