Decomposition Structures for Soft Constraints Evaluation Problems

Laura Bussi

26/07/2019

IN SUPREMÆ DIGNITATIS
1343
UNIVERSITÀ DI PISA

Outline

- Introduction of an algebraic framework for representing and solving Soft Constraint Evaluation Problems
  - Syntax inspired by process algebras
- Correspondence between tree decompositions and terms of the SCEP algebra
- Polyadic soft constraints as an interpretation of the given algebra
- Examples of application

Constraint Satisfaction Problems and dynamic programming

- Constraint Satisfaction Problems can be represented as (hyper)graphs
  - Intuitively, given a domain of values, a set of variables and a set of constraints, we want to find an assignment of the variables s.t. all the constraints are satisfied
  - Further operations required to combine or remove constraints
  - CSP solving is NP-complete
- Dynamic programming applies well to CSPs:
  - We can decompose the problem and then repeatedly solve the subproblems
  - Deciding the best decomposition is known as *secondary optimization problem*: NP-complete

Soft constraints

- More flexible than classical constraints
  - Each soft constraint associates a certain assignement to a value in a poset
  - To solve an SCSP we seek for an assignement of the variable s.t. the total amount is minimized or maximized, due to the fact that we express negative or positive preferences
- SCSPs are difficult to represent and compose
  - Need for a generalisation which allows for representing both the problem's structure and the solution process via an algebraic framework
  - Decomposition and solving are thus correct by construction

Algebraic structures for soft constraints

- Absortive semirings (also known as c-semirings) $\langle S, \oplus, \otimes, 0, 1 \rangle$
  - The $\oplus$ operator is idempotent and 1 is its absorbing element
  - The $\otimes$ operator is associative and commutative
- $\oplus$ is often used to induce the order on the elements of such a semiring
  - We have $a \leq b \iff a \oplus b = b$
- We use $\otimes$ to combine constraints
- We may additionaly require some kind of "subtraction"
  - Based on residuation theory
  - The residuation operator is a kind of weak inverse of $\otimes$

Algebraic structures for soft constraints

- Properties of semirings are useful for manipulating soft constraints
  - Associativity and commutativity of $\otimes$ guarantee the order of combination to be irrelevant
  - Absortivity enforces the fact that adding constraints decreases the number of solutions
  - The existence of the 0 element represents the crisp feature of total dislike of any solution involving a certain assignement
  - The unit 1 allows for modelling the crisp feature of "indifference" for a certain constraint

Polyadic Soft Constraints

- A formalism to manipulate soft constraints based on polyadic algebras
  - Two families of operators, called *cylindrification* and *polyadic substitution* model variables hiding and parameters passing
  - Allows for a polynomial representation of soft constraints
- This algebra respects both the weak and the strong SCEP specification, thus representing an interpretation of the initial algebra

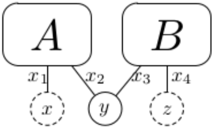
Soft Constraints and concrete networks

- We assume $\mathcal{E}_C$ to be a ranked alphabet of soft constraints, equipped with an arity function $ar : \mathcal{E}_C \to \mathbb{N}$ and a set of variables $\mathbb{V}$
- A *concrete network* is a pair $I \blacktriangleright N$ where
  - $N = (V_N, E_N, a_N, lab_N)$ is a labelled hypergraph over $\mathcal{E}_C$ with no isolated vertices
  - Vertices are a subset of $\mathbb{V}$
  - I is a set of interface variables
- We also assume that $\mathcal{E}_C$ has a function $var : \mathcal{E}_C \to \mathbb{V}^*$ which assigns a tuple of distinct canonical variables to each constraint, i.e. $var(A) \cap var(B) = \emptyset$ if $A \neq B$

Example

- A, B constraints with $ar(A) = ar(B) = 2$
- $var(A) = (x_1, x_2)$, $var(B) = (x_3, x_4)$
- $V_N = \{x, y, z\}$, $E_N = \{e_1, e_2\}$
- $a_N(e_1) = \langle x, y \rangle$, $a_N(e_2) = \langle y, z \rangle$
- $lab_N(e_1) = A$, $lab_N(e_2) = B$

- The concrete network $\{y\} \blacktriangleright N$:



Soft Constraint Satisfaction Problems

- A Soft Constraint Satisfaction Problem (SCSP) is a tuple $(I \blacktriangleright N, \mathbb{D}, S, val)$
- A *value* val_A is a function giving a cost in S to each assignement of canonical variables in A
  - We will use val_e as a function giving a cost to every assignement to variables e is attached to
- Variables I are those of interest
- The solution is a function $sol : (I \to \mathbb{D}) \to S$: for each assignement $\rho : I \to \mathbb{D}$

$$sol(\rho) = \bigoplus_{\{\rho' : V_N \to \mathbb{D} | \rho' \downarrow_I = \rho\}} \left(\bigotimes_{\{i | e_i \in E_N\}} val_e(\rho' \downarrow_{a_N(e_1)}) \right)$$

A canonical representative for networks

- Since we are interested only in interface variables, we can define isomorphisms between networks
- Then two networks are isomorphic if there is an isomorphism between them preserving the interface variables
- An abstract network $I \triangleright N$ is an isomorphism class of concrete networks
- In the following we assume the choice of a canonical representative of abstract networks

Tree decomposition

- A decomposition of a graph can be represented as a tree decomposition
  - Each vertex is a piece of the graph
- A (rooted) tree decomposition of a hypergraph G is a pair $\mathcal{T} = (T, X)$ where T is a rooted tree and $X = \{X_t\}_{t \in V_T}$ is a family of subset of V_G s.t.
  - For each vertex $v \in V_G$ there exists a vertex of $\mathcal{T}$ s.t. $v \in X_t$
  - For each hyperedge $e \in E_G$ there is a vertex of $\mathcal{T}$ s.t. $a_G(e) \subseteq X_t$
  - Each subtree generated from $v \in V_G$ is a rooted tree

Tree decomposition

- The decomposition of a network $I \triangleright N$ is a decomposition of N where we choose as a root a vertex of $\mathcal{T}$ containing all the interface variables
- We'll provide a translation from tree decompositions to SCEP (weak) terms: this will enable applying algebraic techniques to tree decompositions
- We'll refer to *completed versions* of tree decompositions, which explicitly associate components of N to vertices of $\mathcal{T}$
  - for each $v \in V_N (e \in E_N)$, $t_v (t_e)$ is the vertex closest to the root of $\mathcal{T}$ s.t. $v \in X_{t_v}$

SCEP signature

- SCEP algebras are *permutation algebras*
- The SCEP-signature (*s-signature*) is given by the following grammar:

$$p, q := p \,||\, q \mid (x)p \mid p\pi \mid A(\widetilde{x}) \mid nil$$

- The free variables of p ($fv(p)$) are defined recursevly as expected
  - the restriction (x) is a bounding operator
  - the free variables of $p\pi$ are just $\pi(fv(p))$

Strong SCEP specification

- A certain set of axioms toghether with the signature give the strong SCEP specification (*s-specification*)
  - $||$ forms a commutative monoid
  - Restrictions can be swapped and α-converted
  - Permutations distribute over syntactic operators

$(\mathbf{AX}_{\|})$

$$p \parallel q \equiv_s q \parallel p \qquad (p \parallel q) \parallel r \equiv_s p \parallel (q \parallel r) \qquad p \parallel \mathtt{nil} \equiv_s p$$

$(\mathbf{AX}_{(x)})$

$$(x)(y)p \equiv_s (y)(x)p \qquad (x)\mathtt{nil} \equiv_s \mathtt{nil}$$

$(\mathbf{AX}_{\alpha})$

$$(x)p \equiv_s (y)p[x \mapsto y] \qquad (y \notin fv(p))$$

$(\mathbf{AX}_{SE})$

$$(x)(p \parallel q) \equiv_s (x)p \parallel q \qquad (x \notin fv(q))$$

$(\mathbf{AX}_{\pi})$

$$p\,\mathsf{id} \equiv_s p \qquad (p\pi')\pi \equiv_s p(\pi \circ \pi')$$

$(\mathbf{AX}^{p}_{\pi})$

$$A(x_1, \ldots, x_n)\pi \equiv_s A(\pi(x_1), \ldots, \pi(x_n)) \qquad \mathtt{nil}\,\pi \equiv_s \mathtt{nil} \qquad (p \parallel q)\pi \equiv_s p\pi \parallel q\pi$$

$$((x)p)\pi \equiv_s (\pi(x))(p\pi)$$

Weak SCEP specification

- The axiom AX_{SE} in the s-specification states that the scope of restricted variables can be narrowed to terms where they occur free
- This cause s-terms with different decompositions to be equivalent
- To distinguish different decompositions, we define a *weak* SCEP specification (*w-specification*) where AX_{SE} is replaced with

$$(AX^{w}_{(x)}) \quad (x)p \equiv_w p \quad (x \notin fv(p))$$

Normal and canonical form

- W-terms can be seen as networks having a hierarchical structure
- Normal and canonical forms are those of interest
- A w-term is said to be in normal form whenever it is of the form $(\widetilde{x})(A_1(\widetilde{x_1}) \mid\mid \ldots \mid\mid A_n(\widetilde{x_n}))$
- A w-term is said to be in canonical form whenever is obtained by the repeated application of (AX_{SE}) until termination
- As we'll see, a term's form have an impact on its evaluation complexity

Soundness and completeness of networks

- The s-specification is sound and complete w.r.t. networks
- We can define a translation function which, given a concrete network $I \blacktriangleright N$, gives us the corresponding s-term and viceversa
  - $term(I \blacktriangleright N) = (V_N \setminus I)(A_1(\widetilde{x_1}) \mid\mid \dots \mid\mid A_n(\widetilde{x_n}))$
  - $net(p) = fv(p) \blacktriangleright N_p$
- Completeness follows by the fact that if $net(n_1) \cong net(n_2)$, then $n_1 \equiv_s n_2$

SCSPs as SCEPs

- SCSPs can be represented and solved as SCEPs
- SCEPs are indeed more general than SCSPs
  - There exist optimisation problems which are SCEPs and cannot be represented as SCSPs
- Thus we can define an algebra $\mathcal{V}$ s.t. $[\![I \triangleright N]\!]^{\mathcal{V}}$ gives the solution of the SCSP defined over the network

Theorem Given an SCSP with underlying network $I \triangleright N$ and value functions val_A, we have that $I \triangleright N$ evaluated in $\mathcal{V}$ ($[\![I \triangleright N]\!]^{\mathcal{V}}$) is its solution

Evaluation Complexity

$$\langle\!\langle p \parallel q \rangle\!\rangle = \max\{\langle\!\langle p \rangle\!\rangle, \langle\!\langle q \rangle\!\rangle, |fv(p \parallel q)|\} \qquad \langle\!\langle (x)p \rangle\!\rangle = \langle\!\langle p \rangle\!\rangle \qquad \langle\!\langle p\pi \rangle\!\rangle = \langle\!\langle p \rangle\!\rangle$$
$$\langle\!\langle A(\tilde{x}) \rangle\!\rangle = |set(\tilde{x})| \qquad \langle\!\langle \mathtt{nil} \rangle\!\rangle = 0$$

- We now introduce a notion of complexity of w-terms ($\langle\langle - \rangle\rangle$)
- The complexity of p ($\langle\langle p \rangle\rangle$) is the maximum "size" of elements of an algebra $\mathcal{A}$ computed while inductively constructing $[\![p]\!]^{\mathcal{A}}$

Theorem Given a term p, let n be its normal form. For all canonical forms c of p we have $\langle\langle c \rangle\rangle \leq \langle\langle n \rangle\rangle$

Tree decompositions as w-terms

- Given a network $I \triangleright N$, let $\mathcal{CT} = (\mathcal{T}, \{t_x\}_{x \in E_N \cup V_N})$ the completed version of one of its tree decompositions
- We translate $\mathcal{CT}$ into a w-term:
  - Given $t \in T$, let $V(t) = \{v \in V_N \mid t_v = t\}$ and $E(t) = \{e \in E_N \mid t_e = t\}$
  - Suppose t has children $t_1, ..., t_n$ and $E(t))\{e_1, ..., e_k\}$
  - Let $\widetilde{x} = V(t) \setminus I$
- The term $\chi(t)$ is inductively defined as:

$$\chi(t) = (\widetilde{x})(A_1(\widetilde{x_1}) \mid\mid ... \mid\mid A_k(\widetilde{x_k}) \mid\mid \chi(t_1) \mid\mid ... \mid\mid \chi(t_n))$$

Tree decompositions as w-terms

- Given a tree decomposition of $\mathcal{T}$ rooted in r, the corresponding w-term $wterm(\mathcal{T})$ is $\chi(r)$ computed on the completed version of $\mathcal{T}$
- $wterm(\mathcal{T})$ correctly represents the network $\mathcal{T}$ decomposes: if $\mathcal{T}$ is a tree decomposition for $I \triangleright N$, then $[\![wterm(\mathcal{T})]\!]^{\mathcal{N}} = I \triangleright N$
- Furthermore, given a tree decomposition $\mathcal{T}$, we have $\langle\langle wterm(\mathcal{T})\rangle\rangle \leq width(\mathcal{T})$

An algorithm for computing canonical decomposition

- Based on *bucket elimination*
  - Differently from it, the introduced algorithm produces all and only canonical terms
- Here putting a constraint in the bucket of its last variable means to apply the (AX_{SE}) axiom
- The input are an s-term $(R)A$ in normal form and a total order O_R over R
- The output is a w-term P in canonical form

Theorem C is a canonical form of $R(A)$ if and only if there is O_R^C s.t. the algorithm with inputs $(R)A$ and O_R^C outputs C

An algorithm for computing canonical decomposition

Inputs: s-term $(R)A$ in normal form; a total order O_R over R.
Output: w-term P in canonical form.

$P \leftarrow (R)A$
while $O_R \neq \emptyset$
    $x \leftarrow \text{extract max}\, O_R$
    $O_R \leftarrow O_R \setminus \{x\}$
    find all terms $A' \subseteq A$ such that $x \in fv(A')$
    if $A' = \{(R')P'\}$ where P' has no top-level restriction
        $Q \leftarrow$ call the algorithm on $(x)P'$ with order $\{(x,x)\}$
        $P'' \leftarrow (R')Q$
    else $P'' \leftarrow (x)A'$
    $P \leftarrow (R \setminus \{x\})A \setminus A' \cup \{P''\}$
return P

PSC as an interpretation of the initial algebra

- We define the algebra $\mathcal{P}$ of polyadic soft constraints
- Constants are $A^{\mathcal{P}}(x_1, \ldots, x_n)\eta = c_{(x_1,\ldots,x_n)}(\eta \circ \widehat{\sigma})$, $nil^{\mathcal{P}} = 0$
- Operations are
  - $((X)^{\mathcal{P}} c)\eta = (\exists_X c)\eta = \bigvee_\rho \{c\rho \mid \eta_{|\mathbb{V}\setminus X} = \rho_{|\mathbb{V}\setminus X}\}$
  - $c\pi^{\mathcal{P}} = (s_\pi c)\eta = c(\eta \circ \pi)$
  - $c_1 \;||^{\mathcal{P}}\; c_2 = (c_1\eta) \otimes (c_2\eta)$
- $\widehat{\sigma}$ is used to map $var(A)$ to $\langle x_1, \ldots, x_n \rangle$

Weak and strong axioms for $\mathcal{P}$

- Axioms for α-conversion and swapping of restrictions hold in $\mathcal{P}$, as well as those involving permutations and parallel composition
- $\mathcal{P}$ is a w-algebra: it holds $(\exists_X c)\eta = c\eta$ if $X \cap supp(c) = \emptyset$ $(AX^{w}_{(x)})$
- $\mathcal{P}$ is an s-algebra: it holds $X \cap supp(c2) = \emptyset \implies (\exists_X (c_1 \otimes c_2))\eta = (\exists_X (c_1 \otimes \exists_X c_2))\eta = (\exists_X c_1 \otimes \exists c_2)\eta = (\exists_X c_1)\eta \otimes (\exists_X c_2)\eta = (\exists_X c_1)\eta \otimes c_2\eta$

Example

- Consider a network where meeting activities for a group require the existence of paths between every pair of collaborators
- We assume that the network is composed of end-to-end two-way connections with independent probabilities of failure
- We want to find the probability of a certain group to stay connected
- We consider a graph with probabilities associated to each edge: the solution is the probability of some interface vertices staying connected

Example

- We evaluate networks $I \triangleright N$ into an algebra of probability distributions on the partitions of I ($Part(I)$)
- If we are interested in a group J, we compute the probability distribution P for $J \triangleright N$ and then we select $P(\{J\})$
- Let us focus on parallel composition, defined as:

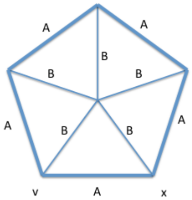
$$[\![I_1 \triangleright N_1 \mid\mid I_2 \triangleright N_2]\!]^{\mathcal{D}} \Pi = \sum_{\Pi' \in Part(I \cup x) \mid \Pi' - x = \Pi} [\![I_1 \triangleright N_1]\!]^{\mathcal{D}} \Pi_1 \times [\![I_2 \triangleright N_2]\!]^{\mathcal{D}} \Pi_2$$

- Here $\Pi \in Part(I_1 \cup I_2)$ and each Π_1, P_2 must belong to $Part(I_1)$ and $Part(I_2)$ respectively. The $\cup$ operator produces the finest partition coarser than the two components and $\times$ is the multiplication on reals

Formal specification

- Considered networks are wheels $W_k(v, x)$
- For $k = 2$ we have the following graph:

$$
\begin{aligned}
R_0(x,y,z) &= A(x,y) \parallel B(x,z) \\
R_{i+1}(x,y,z) &= (v)(R_i(x,v,z) \parallel R_i(v,y,z)) \\
W_k(v,x) &= (z)(R_k(x,v,z) \parallel A(v,x) \parallel B(v,z)) \\
FW_k(v,x) &= (z)(R_k(x,v,z) \parallel B(v,z))
\end{aligned}
$$



Non existence of an SCSP formulation

- The given SCEP does not fit the SCSP format
- Suppose to define the problem as an SCSP:
  - a partition in $Part(I)$ can be represented as an assignment of variables I
  - the solution is the probabilty $sol(\Pi)$ associated to a partition Π
- The solution in the SCSP case, for any two networks, is $val_{N_1}(\Pi_1) \otimes val_{N_2}(\Pi_2)$
  - The solution considers only the probabilities caused by the same Π on the two subnetworks
- This is not compatible with the given definition of parallel composition

References

- *U. Montanari, M. Sammartino, A. Tcheukam* - Decomposition Structures for Soft Constraint Evaluation Problems: An Algebraic Approach
- *F. Bonchi, L. Bussi, F. Gadducci, F. Santini* - Polyadic Soft Constraints

Conclusions

- We saw an algebraic framework for defining and solving SCEPs
- We also saw as polyadic soft constraints can be seen as an s-algebra or a w-algebra
- Using these frameworks, we can apply dynamic programming to SCSPs, thus improving their complexity