TCP-Compose* – A TCP-Net Based Algorithm for Efficient Composition of Web Services Using Qualitative Preferences*

Ganesh Ram Santhanam, Samik Basu, and Vasant Honavar

Department of Computer Science, Iowa State University, Ames IA 50011, USA
{gsanthan,sbasu,honavar}@cs.iastate.edu

Abstract. In many practical applications, trade-offs involving non-functional attributes e.g., availability, performance play an important role in selecting component services in assembling a feasible composition, i.e., a composite service that achieves the desired functionality. We present TCP-Compose*, an algorithm for service composition that identifies, from a set of candidate solutions that achieve the desired functionality, a set of composite services that are *non-dominated* by any other candidate with respect to the user-specified qualitative preferences over non-functional attributes. We use TCP-net, a graphical modeling paradigm for representing and reasoning with qualitative preferences and importance. We propose a heuristic for estimating the preference ordering over the different choices at each stage in the composition to improve the efficiency of TCP-Compose*. We establish the conditions under which TCP-Compose* is guaranteed to generate a set of composite services that (a) achieve the desired functionality and (b) constitute a *non-dominated* set of solutions with respect to the user-specified preferences and tradeoffs over the non-functional attributes.

1 Introduction

Service-oriented computing [1,2,3] offers a powerful approach to assemble complex distributed applications from independently developed software components in many application domains such as e-Science, e-Business and e-Government. Consequently, there is a growing body of work on specification, discovery, selection, and composition of services. The focus of this paper is on service composition, i.e., the problem of assembling a composite service (goal service) from component services from *functional* and *non-functional* specifications.

Functional requirements specify the desired goal service functionality. Barring a few notable exceptions [4,5,6,7], much of the work on service composition has focused on algorithms for assembly of composite services from functional specifications. Some of the major approaches to service composition based on

* This work is supported in part by NSF grants CNS0709217, CCF0702758 and IIS0711356.

A. Bouguettaya, I. Krueger, and T. Margaria (Eds.): ICSOC 2008, LNCS 5364, pp. 453–467, 2008.

functional specifications include: AI planning [8,9,10,11], labeled transition systems [12,13,14], Petri nets [15], among others. (The interested reader is referred to [16,17,18] for surveys).

Non-functional requirements refer to aspects such as security, reliability, performance, and cost of the goal service. For example, among the composite services that achieve the desired functionality, a user might prefer a more secure service over a less secure one; or one with a lower cost over one with a higher cost. Such preferences may be *quantitative* or *qualitative*. In many settings, a user might need to trade off one non-functional attribute against another (e.g., performance against cost); In others, it might be useful to assign relative importance to different non-functional attributes (e.g., security being more important than performance). Hence, there is an urgent need for principled methods that incorporate consideration of user-specified preferences with respect to the non-functional attributes, and the relative importance of the different non functional attributes. Of particular interest are algorithms that ensure that a set of solutions generated constitute a *non-dominated set*. We say that a set N of composite services is a *non-dominated set* if there is no composite service that is not in N that is strictly preferred over one or more of the composite services in N with respect to a set of user-specified preferences over non-functional attributes (and their relative importance).

Against this background, we present a procedure, `TCP-Compose`* for generating, given (i) a set of functional specifications; (ii) a set of preferences with respect to non-functional attributes and their relative importance; (iii) a repository of candidate services with specified input-output behaviors and non-functional attributes; and (iv) *any* sound algorithm for assembling, from a repository of component services: a set of composite services that (a) achieve the desired functionality and (b) are non-dominated with respect to the user-specified preferences over non-functional attributes by any other composite service in the solution set of the algorithm used for functional specification based service composition.

`TCP-Compose`* makes use of Tradeoff-enhanced Conditional Preference Network (TCP-net) [19], a variant of Conditional Preference Network [20], a framework for representing and reasoning with qualitative preferences. *CP*-net provides a compact representation of user-specified preferences with respect to non-functional attributes, by taking advantage of the independence or conditional independence of user preferences with respect to an attribute from preferences with respect to other attributes. *TCP*-net extends the CP-net framework by allowing the specification of the *relative importance* of different attributes (e.g., security is more important than cost).

`TCP-Compose`* uses a heuristic estimate of the preference ordering of alternative partial solutions to a service composition problem that corresponds to different choices of each component service, to improve the efficiency of search for a set of non-dominated solutions. We establish the conditions under which the proposed algorithm is guaranteed to find a set of non-dominated compositions with respect to user-specified qualitative preferences over possible values of each non-functional attribute and the relative importance of different non-functional attributes.

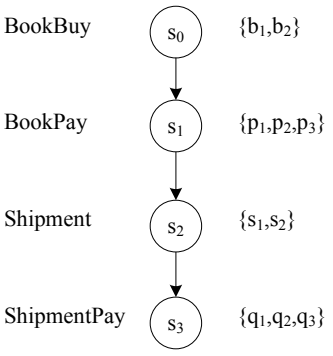

Fig. 1. Goal Service

The rest of the paper is organized as follows: Section 2 introduces the problem of service composition from user-specified functional and non-functional specifications; Section 3 describes the key aspects of CP-net and TCP-net formalisms used for representing and reasoning about user preferences with respect to the non-functional attributes and their relative importance, and outlines the application of TCP-nets to guide service composition based on non-functional requirements; Section 4 describes the algorithm TCP-Compose* and establishes the conditions under which TCP-Compose* is guaranteed to find the set of non-dominated compositions with respect to user-specified qualitative preferences over possible values of each non-functional attribute and the relative importance of different non-functional attributes; Section 5 concludes with a summary, discussion of related work, and an outline of some directions for further research.

2 Problem Specification

We introduce the problem of service composition from user-specified functional and non-functional requirements using a simple example. Suppose a user is interested in assembling a goal service G shown in Fig. 1 from a repository of services $R = \{b_1, b_2, p_1, p_2, p_3, s_1, s_2, q_1, q_2, q_3\}$–where b_i's are book buying services, s_i's are shipment services and p_i's and q_i's are payment services that can work with b_i's and s_i's respectively. Suppose $(p_3, q_2), (b_2, s_2), (b_2, q_2), (b_2, q_3)$ are functionally incompatible and hence cannot be used together in any valid composition. The goal service should allow the user to buy book(s) from an online store, pay the store through a credit card service, arrange for shipping the book through a shipment service and pay for the shipping. In Fig. 1, each of the steps in the goal service is annotated with the set of services from the repository that provide the respective functionality.

What we have so far is an informal specification of a service composition task based on user-specified functional requirements. We now turn to specification of user preferences with respect to three non-functional attributes: reliability, security, and availability of the goal service denoted by R, S, and A respectively.

Table 1. Domain Definition

Preference Variable	Domain of Preference Variable
Reliability(R)	$\{L_R, H_R\}$
Security (S)	$\{L_S, M_S, H_S\}$
Availability(A)	$\{L_A, H_A\}$

Fig. 2. Example CP-net and TCP-net

Suppose the available services can have Low (L_R) or High (H_R) reliability; Low (L_S), Medium (M_S) or High (H_S) security; and Low (L_A) or High (H_A) availability as shown in Table 1. Assume that the following non-functional attributes are known of each of the component services: $b_1 : L_R, b_2 : H_R, p_1 : L_S, p_2 : M_S, p_3 : H_S, s_1 : L_A, s_2 : H_R, q_1 : L_S, q_2 : M_S, q_3 : H_S$.

Now suppose that the user's preference with respect to security level is *not* independent of the reliability of the service. Suppose further that when the reliability is low, the user prefers high security; and when reliability is high the user is willing to settle for lower security (say, because of the prohibitive cost of achieving both high security and reliability); Suppose further that the user prefers high availability to low availability irrespective of the reliability and security of the service. Such information can be represented concisely using a CP-net with three nodes denoting the three attributes R, S, and A. The single headed arrows (e.g., from R to S) denote dependence among user preferences with respect to the attributes under consideration. The qualitative preferences of the user with respect to each attribute (conditioned on the preferences over attributes that such preference is dependent on) are specified by the conditional preference table (CPT) that annotate each node (Fig. 2(a)). Suppose further that the user attaches greater importance to availability relative to security. Such assertions of relative importance of one attribute over another are represented using double headed arrows in TCP-net shown in Fig. 2(b). The information regarding preferences with respect to R, S, and A in Fig. 2(b) are the same as those shown in Fig. 2(a).

Given the preferences with respect to the non-functional attributes and their relative importance, our task is to identify from the solution space, i.e., the set of composite services that satisfy the user-specified functional requirements, a subset that forms a *non-dominated* set with respect to a set of user-specified

preferences over non-functional attributes (and tradeoffs among them) that are captured by a TCP-net. It should be noted that a unique optimal composition exists only when the corresponding TCP-net induces a total ordering over the set of candidate feasible composite services, that is, the set of composite services that satisfy the user-specified functionality. TCP-Compose* does not assume the existence of a total order induced by the TCP-net over user-specified preferences and relative importance among attributes. Instead, we return a set of feasible composite services that constitute a non-dominated set with respect to the TCP-net that reflects the users preferences and tradeoffs with respect to the non-functional attributes.

3 Representing Preferences Using CP-Nets and TCP-Nets

We first introduce the basic notions of preference relation, preferential independence under the *ceteris paribus*[1] semantics and the notion of relative importance among variables. We start with a set of preference variables $V = \{X_1, \ldots X_n\}$ with finite domains $D(X_1), \ldots D(X_n)$.

An *outcome* o is a complete assignment of all variables X_i in V. The set of outcomes is $O \subseteq D(X_1) \times D(X_2) \times \ldots \times D(X_n)$. A *preference ranking* is a total preorder over the set of outcomes O. We denote the fact that outcome $o_1 \in O$ is *at least as preferred (strictly preferred)* to outcome $o_2 \in O$ by $o_1 \succeq o_2 (o_1 \succ o_2)$. We denote the fact that the user is *indifferent* between outcomes o_1 and o_2 by $o_1 \cong o_2$ if neither $o_1 \succeq o_2$ nor $o_2 \succeq o_1$.

Preferential Independence. In order to understand the need for preferential independence, we note that the set of possible outcomes is exponential in the number of preference variables n (where $n = |V|$). Further, the set of possible total preorders is doubly exponential in n. A set of variables $X \subseteq V$ is *preferentially independent* of $Y = V - X$ if for all possible values of Y, the *preference order* among various assignments to X is the same. Formally, a set of variables X is *preferentially independent* of the set of variables $Y = V - X$ iff for all $x_1, x_2 \in D(X)$; $y_1, y_2 \in D(Y)$ (where we use $D(\cdot)$ to denote the domain of set of variables also), we have: $x_1y_1 \succeq x_2y_1$ *iff* $x_1y_2 \succeq x_2y_2$. We say that x_1 is preferred to x_2 *ceteris paribus* (all else being equal).

Conditional Preferential Independence. Let X, Y, Z be a partition of V and let $x_1, x_2 \in D(X)$; $y_1, y_2 \in D(Y)$ and $z \in D(Z)$. X and Y are *conditionally preferentially independent* of each other given z *iff*, $\forall x_1, x_2, y_1, y_2$ we have: $x_1y_1z \succeq x_2y_1z$ *iff* $x_1y_2z \succeq x_2y_2z$.

Relative Importance. In Fig. 2(a), we observe that the variables availability and reliability are preferentially independent. Thus, the CP-net, does not assert whether an outcome with high availability and low reliability is preferred to one with low availability and high reliability: all we know from the CP-net is that higher availability and higher reliability are preferred. If we have the additional

[1] Ceteris paribus is a Latin phrase that means "all other things being equal".

information that although reliability and availability are preferentially independent, reliability is *more important* to the user than availability, we can infer that given a choice, the user would settle for lower availability instead of compromising on reliability. Formally, let X and Y be a pair of preferentially independent variables given $V - \{X, Y\}$. We say that X is *relatively more important* than Y, denoted by $X \triangleright Y$, if

$$\forall w.\ w \in D(W),\ where\ W = V - \{X, Y\}, \forall x_1, x_2 \in D(X), \forall y_a, y_b \in D(Y)$$
$$x_1 \succ x_2 \Rightarrow x_1 y_a w \succ x_2 y_b w.$$

Note that the preference $x_1 y_a w \succ x_2 y_b w$ holds even if $y_b \succeq y_a$, since any change for the worse in Y is preferred to any change for the worse in X. A conditional version of relative importance is defined analogously as follows. Let X and Y be a pair of preferentially independent variables given $V - \{X, Y\}$ and $z \in D(Z)$. We say that X is *conditionally relatively more important* than Y given z, denoted by $X \triangleright_z Y$, if the following holds:

$$\forall w.\ w \in D(W),\ where\ W = V - (\{X, Y\} \cup Z),\ Z \subseteq W$$
$$\forall x_1, x_2 \in D(X), \forall y_a, y_b \in D(Y) :\ (x_1 \succ x_2\ given\ zw)\ \Rightarrow x_1 y_a z w \succ x_2 y_b z w.$$

3.1 TCP-Nets

TCP-nets [21,19], extend the CP-net representation by incorporating the *relative importance* among pairs of attributes. The nodes of a TCP-net are the preference attributes V, and there are three types of edges. The first type of edge is a directed edge (single headed arrow) from X to Y used to model preferential dependence of Y on X. Such an edge asserts the preferential dependence of an attribute X_i on the assignment of its parents $Pa(X_i)$. Each node (preference attribute) X_i that has a non empty set of parents $Pa(X_i)$ influencing its preferences is annotated with the conditional preference relation called conditional preference table $CPT(X_i)$. More formally, for each assignment of $Pa(X_i)$, $CPT(X_i)$ specifies a total order over $D(X_i)$. The second type of edge is a double headed arrow which captures the relative importance among a pair of attributes, i.e. if there is such an edge from X to Y then X is relatively more important than Y. The third type of edge is an undirected edge which captures the conditional relative importance between X and Y given Z. We refer to [19] for formal definitions of TCP-nets.

Definition 1 (Completion). *[19] The* completion *of a partial assignment z is defined as a complete assignment or an outcome consistent with z, denoted $Comp(z)$. By consistency, we mean that if a preference attribute X_i has a valuation v_i in z then the valuation of X_i is also v_i in $Comp(z)$.*

Definition 2 (Most Preferred Completion). *[19] The* most preferred completion *of a partial assignment z, denoted $PrefComp(z, \mathcal{N})$ is defined as a* completion *of z that is preferentially optimal with respect to the TCP-Net $\mathcal{N}$, i.e. $\nexists\ o \in O :\ o \succ PrefComp(z, \mathcal{N})$ such that o is a completion of z and consistent with z.*

Remarks

1. We restrict our discussion to the class of *conditionally acyclic* TCP-nets that have been shown to be satisfiable with respect to a preference relation [19].
2. Given a *conditionally acyclic* TCP-net, it is possible to order the set of all outcomes O [19]. In other words, there exists a total order (that can be obtained using a topological sort) of the set of outcomes O that is *consistent with* the given TCP-net. However, several orderings of O can be consistent with a given conditionally acyclic TCP-net. For example, in a total preorder, there could be an outcome o such that $\nexists o' \succ o$ with respect to $\mathcal{N}$, but one cannot define o as the unique most preferred outcome. In our example, considering tuples of valuations of the non-functional attributes of a service, if the user did not give the information that R is relatively more important than A, then we would not be able to assert a preference among compositions with outcomes $o_1 = (H_R, L_S, H_A)$ and $o_2 = (L_R, H_S, L_A)$ (where subscripts denote the corresponding non-functional attributes reliability (R), security (S) and availability (A)). In this case, the user may like the composition system to return both the compositions if both o_1 and o_2 are non-dominated, i.e., $\nexists o' \succ o_1$ and $\nexists o' \succ o_2$. The algorithm we present, TCP-Compose* guarantees that in the absence of a *unique total order* over the outcomes, the outcome corresponding to each composition in the solution set is non-dominated by the outcome corresponding to any other feasible composition.
3. We also note that there is another variant of the TCP-net, known as UCP-nets [19] that capture quantitative preferences and relative importance information using utility functions. However, since we are not dealing with quantitative preferences, we stick to the basic qualitative TCP-nets.

3.2 Utilizing TCP-Nets in Web Service Composition

We now proceed to describe how TCP-nets can be used to model qualitative preferences during Web service composition. For this we will use *dominance queries* [19] of the form $o \overset{?}{\succ} o'$ with respect to $\mathcal{N}$ (in other words whether o is preferred to or dominates o'). The problem of Web service composition is to assemble a composite service that achieves a desired functionality from a set of component services. More precisely, we have:

Definition 3 (Web service composition problem). *Given a target or goal service G and a repository of available services $R = \{W_1, W_2 \ldots W_n\}$, Web service composition amounts to creating a set of composite services $C = \{C_1, C_2 \ldots C_m\}$ such that $\forall i \leq m, C_i = W_{i_1} \oplus W_{i_2} \ldots \oplus W_{i_k}$ and $\forall l \leq i_k, W_l \in R$ such that C_i is functionally equivalent [2] to the G, denoted by $C_i \equiv G$. In the above, $\oplus$ is the* composition *operator for composing two services.*

[2] Functional equivalence can be defined in many ways depending on the particular formalism used to describe the services. For example, if labeled transition systems are used for describing the services, checking the functional equivalence of a composite service to a goal service reduces to checking the *bisimulation equivalence* of the corresponding labeled transition systems [12,13].

Note that $\oplus$ is a *generic* composition operator and $W_{i_1}, W_{i_2} \ldots W_{i_k}$ is an arbitrary ordering of the components in C_i such that W_{i_j} is selected before $W_{i_{j+1}}$ in constructing C_i. We now proceed to describe an approach for using the TCP-net representation of user-specified *non-functional* requirements to guide service composition using any of the standard methods that can generate compositions that satisfy user-specified *functional* requirements.

4 TCP-Compose*

We present an algorithm, TCP-Compose*, that uses a preference guided heuristic to come up with the most preferred compositions among the candidates.

4.1 Search Space of TCP-Compose*

We cast the problem of assembling from a set of available component services, a composite service with the desired functionality as a state space search problem. The empty composition $\perp$ is the start state; the set of *feasible extensions* using one of the available components from any given state define the successors of that state; and the set of feasible candidate compositions correspond to the goal states. The cost function at any state is given by the *preference valuation* of the partial composition corresponding to that state.

Definition 4 (Feasible Extension). *A feasible extension to a partial composition P is defined as a partial composition $P' = P \oplus W_i, W_i \in R$ such that the partial composition P' is functionally equivalent to a part of the goal service.*

Let $\mathcal{N}$ be a TCP-net with a set of preference attributes $V = \{X_1, X_2, \ldots X_p\}$ with finite domains $D(X_1), D(X_2), \ldots D(X_k)$ respectively where each preference attribute corresponds to a non-functional attribute of a composition. We assume that such a TCP-net specification is given by the user as input to the algorithm TCP-Compose*.

Each of the leaf nodes is a goal node and corresponds to valid or feasible candidate composite services that are functionally equivalent to the goal service. Note that the the nodes of the tree may have varying but finite branching factors. Fig. 3 illustrates the search space for our goal service given in Fig. 1 with respect to the TCP-net given in Fig. 2. The shaded nodes correspond to partial compositions that were actually expanded further. The numbers in the boxes next the nodes show the order in which the corresponding nodes are expanded. The nodes that are not shaded are generated but not further pursued by the algorithm: For example, although TCP-Compose* explores partial composition $b_1 \oplus p_1$, its feasible extension $b_1 \oplus p_1 \oplus s_1$ is not explored. The annotation Val denotes the *preference valuation* and β denotes the *most preferred completion* of the partial composition corresponding to each node. They are formally defined below.

Definition 5 (Preference Valuation). Preference valuation *is a function* $F : W \times X \rightarrow \bigcup(D(X_i) \cup \{-\})$, *where* $W = \{W_1, W_2 \ldots W_n\}$, $X = \bigcup X_i$. *The value* $\{-\}$ *denotes that the valuation of the corresponding attribute is unknown. We*

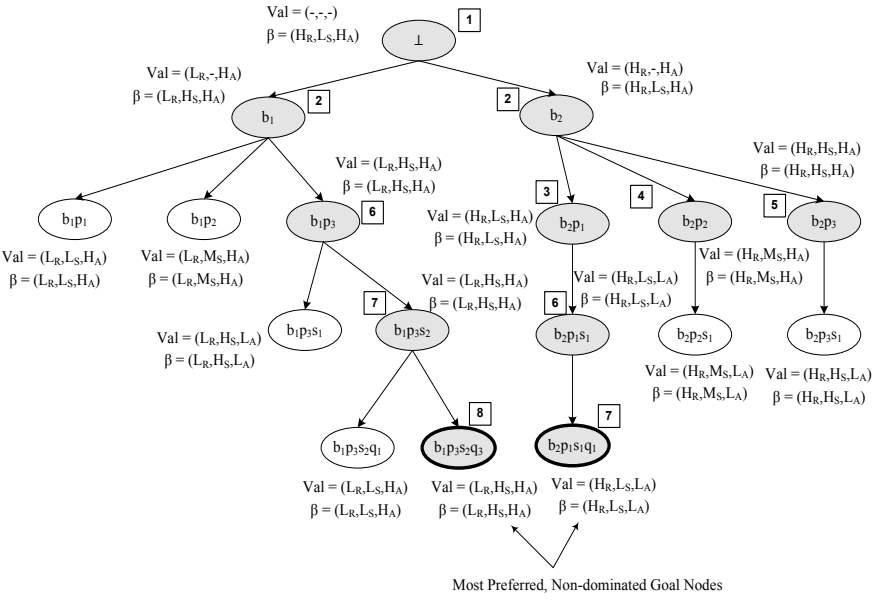

Fig. 3. Search Space for TCP-Compose* when the TCP-net does not induce a total order over the set of valuations

denote the valuation of an attribute X_i in a Web service W as $F(W)(X_i) = v_i, v_i \in D(X_i) \cup \{-\}$. We define the valuation of an attribute X_i in a composition *of two services W_i, W_j as $F(W_i \oplus W_j)(X_i) = F(W_i)(X_i) \odot F(W_j)(X_i)$, where*

$$F(W_i)(X_p) \;\odot\; F(W_j)(X_p) \quad = \quad \begin{cases} F(W_j)(X_p), & F(W_i)(X_p) \succ F(W_j)(X_p) \\ F(W_i)(X_p), & otherwise. \end{cases}$$

The preference valuation of a partial composition $P = W_1 \oplus W_2 \oplus \ldots W_l$ with respect to an attribute X_p is defined inductively as $F(P)(X_p) = F(W_1)(X_p) \odot F(W_2)(X_p) \ldots \odot F(W_l)(X_p)$. We also denote the preference valuation (over all attributes) of a partial composition P as the tuple $Val(P) = (F(P)(X_i), F(P)(X_2) \ldots F(P)(X_k))$.

Thus, the preference valuation of a partial composition with respect to a non-functional attribute corresponds to the least preferred valuation of that attribute among the participating component services in the composition. For example, in Fig. 3 the valuation of the partial composition $b_2 \oplus p_1 \oplus s_1$ with respect to the attribute availability (A) is low (L_A) because the component s_1 has low availability although the component b_2 has high availability.

Definition 6 (Most Preferred Completion). *The* most preferred completion *of a preference valuation of a partial composition P is defined as a complete assignment to all attributes $X_1, X_2, \ldots X_k$, $\beta(P) = PrefComp(Val(F(P)), \mathcal{N})$ where the function $PrefComp$ is as defined in Def.2.*

Algorithm 1. TCP-Compose* $(\mathcal{N}, \varphi, G, R)$

1. Compute heuristic $\rho \leftarrow h(\varphi)$
2. **for all** $P \in \rho$ **do**
3. **if** $(P \equiv G$ and $\nexists\, Q \in \theta : \beta(Q) \succ \beta(P))$ **then**
4. $\theta \leftarrow \theta \cup \{P\}$
5. $\varphi \leftarrow \varphi - \{P\}$
6. **for all** $Q \in \theta$ **do**
7. **if** $\beta(P) \succ \beta(Q)$ **then**
8. $\theta \leftarrow \theta - \{Q\}$
9. **end if**
10. **end for**
11. **else**
12. Find the next set of feasible partial compositions expanding P
$\psi \leftarrow \{P_i \mid P_i = P \oplus W_i, \quad W_i \in R \quad and \quad P \oplus W_i \equiv G\}$
13. $\varphi \leftarrow \varphi \cup \psi - \{P\}$
14. **end if**
15. **end for**
16. **if** $(\varphi = \phi)$ **then**
17. **return** θ
18. **end if**
19. TCP-Compose* $(\mathcal{N}, \varphi, G, R)$

Intuitively, it is easy to see why $\beta(P_i)$ is a heuristic estimate of the most preferred composition that can be realized by extending the given P_i. In our search for compositions, $\beta(P_i)$ denotes the estimate of most preferred feasible candidate composite service that is equivalent to the goal service that is realizable from the current partial composition P_i.

Definition 7 (Heuristic Function h). *We define the heuristic function h as $h: 2^{\mathcal{P}} \rightarrow 2^{\mathcal{P}}$, where $\mathcal{P}$ is a set of partial compositions and $S := h(\varphi)$, $S \subseteq \varphi \subseteq \mathcal{P}$ is such that $\forall P_0 \in S$ and $\forall P_i \in \varphi$, $\beta(P_i) \not\succ \beta(P_0)$. We also define $h(\phi) = \perp$ and $\perp \oplus W_i \equiv W_i$.*

The above definition of the heuristic function h makes it clear that, for any set of partial compositions φ, $h(\varphi)$ is the set of partial compositions whose valuations are non-dominated by the valuation of any other partial composition in φ.

Algorithm 1 shows the listing for TCP-Compose*. The main idea behind TCP-Compose* is to use a best first search strategy to find a set of non-dominated feasible candidate compositions equivalent to the goal service. To guide the search, the algorithm applies the heuristic function h to the set of partial compositions under consideration. The algorithm is initially invoked with the parameters $(\mathcal{N}, \phi, G, R)$. The set of partial compositions under consideration for expansion are maintained in φ, and the algorithm uses $h(\varphi)$ to select the set of non-dominated compositions ρ to expand (line 1). Next, for each of the compositions in the non-dominated set ρ, if there is a candidate composition P which is functionally equivalent to the goal service, then the algorithm adds P to the solution set θ, provided none of the existing solutions already in θ strictly

dominate it (lines 2 - 5). If P now strictly dominates any of the existing solutions in θ then those candidate solutions are removed from θ (lines 6 - 10). For partial compositions that are not candidate compositions in ρ, the algorithm proceeds to compute the set of feasible extensions and adds them to φ (lines 11 and 14). The algorithm terminates with the set of candidate solutions θ if there are no more compositions to explore (lines 16 - 18), and finally, the process is repeated (line 19) until there are no more compositions left to explore.

We now proceed to describe how the algorithm explores the search space when the TCP-net does not induce a total order on the set of non-functional attribute valuations. In the search space illustrated in Fig. 3 in the first run, when expanding the root node there are two possible partial compositions namely b_1 and b_2 respectively. Note that the valuations of partial compositions b_1, b_2 are incomparable with respect to the TCP-net $\mathcal{N}$, and in the second run, both the partial compositions in φ are expanded. In runs 3, 4 and 5 the compositions $b_2 \oplus p_1, b_2 \oplus p_2, b_2 \oplus p_3$ are expanded, as their valuations strictly dominated others in their respective iterations. However, in the sixth run, the algorithm again finds that the non-dominated compositions $b_1 \oplus p_2$ and $b_2 \oplus p_1 \oplus s_1$ are incomparable, and hence expands both. Notice that when expanding $b_2 \oplus p_1 \oplus s_1$, the feasible extensions also have the component s_1 (we assumed that b_2, s_2 are functionally incompatible and cannot be composed together) which has lower reliability and availability, but there is a still a possibility of a service with b_1 ending up with a candidate composition with high availability. In the seventh run, the algorithm identifies $b_2 \oplus p_1 \oplus s_1 \oplus q_1$ as a solution, and finally in the eighth run the algorithm terminates with both the candidate compositions $b_1 \oplus p_3 \oplus s_2 \oplus q_3$ and $b_2 \oplus p_1 \oplus s_1 \oplus q_1$ as the non-dominated candidate compositions. This illustrates how the less preferred compositions like $b_1 \oplus p_1$ are actually not explored by the algorithm, consequently pruning of the search space.

4.2 Properties of TCP-Compose*

We show that the algorithm TCP-Compose* is guaranteed to return the set of composite services each of which is functionally equivalent to the user-specified goal service that constitute a non-dominated set with respect to a set of user preferences over the non-functional attributes.

Lemma 1. *For any partial composition P, $\beta(P) \succeq \beta^\star(P)$, where $\beta^\star$ gives the valuation of the* actual *most preferred feasible composition starting with P.*

Proof. Suppose by contradiction, there exists a partial composition P and a $\beta^\star$ such that $\beta^\star(P) \succ \beta(P)$. This implies that there is a feasible candidate composition C starting from the partial composition P such that $Val(C) \succ \beta(P)$, or there is a sequence of feasible extensions from P to C such that $Val(C) \succ PrefComp(Val(P), \mathcal{N})$ with respect to $\mathcal{N}$, by the definition of $\beta(P)$. This clearly contradicts the definition of $PrefComp(Val(P), \mathcal{N})$. □

Theorem 1. *TCP-Compose* is guaranteed to return the set of feasible composite services that constitute a non-dominated set with respect to a given TCP-net.*

Proof (Sketch, by contradiction)

Suppose TCP-Compose* does not terminate with the set of non-dominated candidate compositions. There are three cases to consider.

1. TCP-Compose* terminates with a set of compositions such that one of the solutions returned by TCP-Compose* is a not feasible composition. This contradicts the Step 3 of the algorithm where the terminating condition is clearly only satisfied for feasible compositions.
2. TCP-Compose* fails to terminate. This is not possible because although the algorithm is recursive, the search tree is finite, and in each iteration, previously examined partial compositions are not revisited.
3. Starting with a partial composition P, TCP-Compose* terminates with a set of candidate compositions such that one of the solutions corresponds to a feasible candidate composition C' with a sub-optimal preference valuation $Val(C')$, i.e. $\beta^\star(P) \succ Val(C')$, where $\beta^\star$ gives the *actual* most preferred feasible composition starting with P.

   By Lemma 1, at each step, $\beta(P) \succeq \beta^\star(P) \succ Val(C') \Rightarrow \beta(P) \succ Val(C')$. So in the last step just before termination, by the definition of the heuristic function h and Line 1 of TCP-Compose*, the algorithm would have chosen to expand the composition P rather than the partial composition that just preceded C'. Hence, the algorithm could not have terminated with any composition C' such that $\beta^\star(P) \succ Val(C')$, and hence it would return only non-dominated candidate compositions.

This proves that TCP-Compose* is guaranteed to return the set of feasible composite services that constitute a non-dominated set with respect to a given TCP-net. □

5 Summary and Discussion

Most of the work on service composition has focused on algorithms for assembling, from a set of available component services, a composite service that achieves the user-specified functionality. However, in many applications, preferences over non-functional attributes e.g., availability, performance as well as tradeoffs among them can influence the choice of the component services in assembling a composite service that achieves the desired functionality. Hence, there is a growing interest in techniques that incorporate such non-functional considerations into service composition. For example, Zeng et al. [4,5] have explored linear programming methods for optimizing the choice of services based on non-functional attributes based on user-specified weights and *utility functions*. Yu et al. [6] have explored a formulation of service composition as a combinatorial optimization (multi-choice multi-dimensional 0-1 Knapsack problem) and as a graph search problem wherein the non-functional constraints are encoded by the edges in the graph. Berbner et al. [7] have proposed a heuristic approach, using simulated annealing and integer programming, for selecting services based on non-functional attributes. Each of these approaches assume a *quantitative*

measure of preference over alternative valuations of non-functional attributes. This is tantamount to assuming the existence of a *cardinal* utility function [22]. Eliciting such a utility function, over multiple not necessarily independent attributes, from users presents a significant challenge in practice. Hence, methods that can utilize *qualitative* information regarding preferences over non-functional attributes are of significant interest.

Against this background, we have presented TCP-Compose*, a procedure for generating, given (i) a set of functional specifications; (ii) a set of preferences with respect to non-functional attributes and their relative importance; (iii) a repository of candidate services with specified input-output behaviors and non-functional attributes; and (iv) *any* sound algorithm for assembling, from a repository of component services: a set of composite services that (a) achieve the desired functionality and (b) are non-dominated with respect to the user-specified preferences over non-functional attributes by any other composite service in the solution set of the algorithm used for functional specification based service composition. TCP-Compose* uses TCP-net, a graphical modeling paradigm for representing and reasoning with qualitative preferences and importance of alternative partial solutions to a service composition problem that corresponds to different choices of each component service. An important feature of TCP-Compose* is that it offers a generic approach to augment *any* of a broad range of available algorithms for assembling feasible composite services that achieve the user-specified functionality with the ability to consider qualitative preferences and tradeoffs over non-functional attributes of the desired goal service.

Schropfer et al. [23] have recently proposed a TCP-net based formulation of qualitative user preferences over non-functional attributes for ranking and selecting *individual* services. In contrast, the focus of TCP-Compose* is on the assembly of a set of *composite* services that constitute a non-dominated set with respect to a set of user-specified preferences and tradeoffs over non-functional attributes.

Shaparau et al. [11] have proposed an algorithm for contingent planning with goal preferences which can also be used for service composition. The planning algorithm requires the user to specify *explicit* preferences over goals. This is tantamount to explicitly specifying an ordering over all feasible composite services. Hence, this approach is likely to be impractical in settings where the number of component services in the repository is large. In contrast, the approach used in TCP-Compose* requires the user to specify only the preferences and tradeoffs over the non-functional attributes that in turn *induce* a preference over the feasible composite services.

Work in progress is aimed at the implementation and experimental evaluation of TCP-Compose* on a range of benchmark problems of varying complexity. Some interesting directions for further research include: investigation of approaches for handling of *global* non-functional constraints (e.g., no composite service which has security level below a specified threshold is acceptable); customized versions of TCP-Compose* that take advantage of specific representations and algorithms used in the search for feasible solutions.

References

1. Bichler, M., Lin, K.J.: Service-oriented computing. Computer 39(3), 99–101 (2006)
2. Papazoglou, M.: Service-oriented computing: concepts, characteristics and directions. In: Proceedings of the Fourth International Conference on Web Information Systems Engineering, pp. 3–12. IEEE Computer Society, Los Alamitos (2003)
3. Huhns, M.P., Singh, M.P.: Service-oriented computing: Key concepts and principles. Internet Computing 9(1), 75–81 (2005)
4. Zeng, L., Benatallah, B., Dumas, M., Kalagnanam, J., Sheng, Q.Z.: Quality driven web services composition. In: Proceedings of the 12th international conference on World Wide Web, pp. 411–421. ACM, New York (2003)
5. Zeng, L., Benatallah, B., Ngu, A.H.H., Dumas, M., Kalagnanam, J., Chang, H.: Qos-aware middleware for web services composition. IEEE Transactions on Software Engineering 30(5), 311–327 (2004)
6. Yu, T., Lin, K.J.: Service selection algorithms for composing complex services with multiple qos constraints. In: Benatallah, B., Casati, F., Traverso, P. (eds.) ICSOC 2005. LNCS, vol. 3826, pp. 130–143. Springer, Heidelberg (2005)
7. Berbner, R., Spahn, M., Repp, N., Heckmann, O., Steinmetz, R.: Heuristics for qos-aware web service composition. In: Proceedings of the IEEE International Conference on Web Services - ICWS 2006, pp. 72–82 (2006)
8. Pistore, M., Traverso, P., Bertoli, P.: Automated composition of web services by planning in asynchronous domains. In: Fifteenth International Conference on Automated Planning and Scheduling, p. 211 (2005)
9. Traverso, P., Pistore, M.: Automated composition of semantic web services into executable processes. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 380–394. Springer, Heidelberg (2004)
10. Sirin, E., Parsia, P., Wu, D., Hendler, J., Nau, D.: Htn planning for web service composition using shop2. Journal of Web Semantics 1(4), 377–396 (2004)
11. Shaparau, D., Pistore, M., Traverso, P.: Contingent planning with goal preferences. In: Proceedings, The Twenty-First National Conference on Artificial Intelligence. AAAI Press, Menlo Park (2006)
12. Pathak, J., Basu, S., Lutz, R., Honavar, V.: Selecting and composing web services through iterative reformulation of functional specifications. In: Proceedings of the 18th International Conference on Tools with Artificial Intelligence, pp. 445–454. IEEE Computer Society, Los Alamitos (2006)
13. Pathak, J., Basu, S., Honavar, V.: On context-specific substitutability of web services. In: Proceedings of the International Conference on Web Services, pp. 192–199. IEEE Computer Society, Los Alamitos (2007)
14. Pathak, J., Basu, S., Honavar, V.: Composing web services through automatic reformulation of service specifications. In: Proceedings of the 5th IEEE International Conference on Services Computing (to appear, 2008)
15. Hamadi, R., Benatallah, B.: A petri net-based model for web service composition. In: Proceedings of the 14th Australasian database conference, pp. 191–200. Australian Computer Society, Inc. (2003)
16. Dustdar, S., Schreiner, W.: A survey on web services composition. International Journal on Web and Grid Services 1(1), 1–20 (2005)
17. Pathak, J., Basu, S., Honavar, V.: Assembling composite web services from autonomous components. In: Maglogiannis, I., Karpouzis, K., Soldatos, J. (eds.) Emerging Artificial Intelligence Applications in Computer Engineering. IOS Press, Amsterdam (in press, 2008)

18. Pistore, M., Marconi, A., Bertoli, P., Traverso, P.: Automated composition of web services by planning at the knowledge level. In: Nineteenth International Joint Conference on Artificial Intelligence, pp. 1252–1259 (2005)
19. Brafman, R.I., Domshlak, C., Shimony, S.E.: On graphical modeling of preference and importance. Journal of Artificial Intelligence Research 25, 389–424 (2006)
20. Boutilier, C., Brafman, R.I., Domshlak, C., Hoos, H.H., Poole, D.: Cp-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. Journal of Artificial Intelligence Research 21, 135–191 (2004)
21. Brafman, R.I., Domshlak, C., Shimony, S.E.: Introducing variable importance tradeoffs into cp-nets. In: Proceedings of Uncertainity in Artificial Intelligence, pp. 69–76 (2002)
22. Keeney, R.L., Raiffa, H.: Decisions with multiple objectives: Preferences and value trade-offs (1993)
23. Schropfer, C., Binshtok, M., Shimony, S.E., Dayan, A., Brafman, R., Offermann, P., Holschke, O.: Introducing preferences over nfps into service selection in soa. In: Non Functional Properties and Service Level Agreements in Service Oriented Computing Workshop (2007)