セマンティック・ウェブサービスの合成問題

Semantic Web Service Composition Problem

小出 誠二*1　武田 英明*2
Seiji Koide　Hideaki Takeda

*1国立情報学研究所総研大　*2国立情報学研究所，東京大学 RACE
National Institute of Informatics, Sokendai　National Institute of Informatics and RACE Univ. Tokyo

The web service composition problem contains two aspects of AI disciplines, i.e., task planning and automatic programming. The semantics of web service composition from the task planning view has been previously reported by authors. The typed unification, which is a basic requisite technology in web service composition, is also reported. In this paper, we focus on the web service semantics from the viewpoint of programming using terms of Hoare Logic. We emphasize that the web service composition is a process of automated programming rather than task planning. Then, the semantics of Hoare Logic is extended so as to be compatible with precondition/effects in task planning based on the description of Hoare Logic semantics by Cousot. Finally, we propose the integration of the unifier in the unification process of Logics and the environment of functional programming so as to obtain the common framework of logic variables and program variables.

1. はじめに

ウェブサービス合成問題は AI プラニング問題の一種と考えられ，これまでその立場にたった研究が行われてきた [Sirin 04, Kuter 05, Sohrabi 06]．しかし，計画問題とウェブ合成問題には一見微妙なしかし本質的な違いがある．第 1 に，ウェブサービスには (計画問題における) 前提条件と効果に加えて，(関数プログラミングと同様な) 入出力がある．しかも，入出力変数が前提条件や効果にある場合には，両者の間に相互作用が起こり，問題をより複雑にする．第 2 に，ウェブサービス合成問題における制約は，計画問題におけるタスクの部分順序制約ではなく，プログラム制御構造でありしかもそこにはデータフローと制御フローが存在する．したがって，ウェブサービス合成問題はサービス発見選択における非決定論的 (non-deterministic) 問題としてはタスク計画の性格を有するが，制約問題としては計画問題というよりも自動プログラミング問題に近い．

Sirin ら [Sirin 04] は，階層型タスクネットワーク (HTN) 計画のシステム SHOP2 [Nau 03] を用いてウェブサービス合成を行ったが，そこでは SHOP2 を用いるために，OWL-S サービス記述から SHOP2 タスク計画領域への自動変換器が開発された．しかし，その変換を可能にするために原子ウェブサービスは情報提供 (効果なし) か世界変更 (出力なし) のどちらかであるとされ，情報提供かつ世界変更 (出力と効果がある) のウェブサービスは対象とされなかった．その理由は，計画問題では前提条件がその時々の状態空間に照らして評価されるが，前提条件中の論理変数がサービス出力に現れた場合，効果という世界変更の副作用を起こすことなく出力を評価することができないからである．

一方，ウェブサービスをこれまで自動プログラミングの立場から取り組んだ例がなかったわけではない．例えば研究初期において，Ankolekar[Ankolekar 04] は Spin [Holzmann 03] というプログラム検証システムを用いてウェブサービス合成を行ったが，そこでは Spin を用いるために前提条件と効果が無視された．そして残念ながらこの立場からのウェブサービス合成に関する研究がその後発展させられることはなかった．

連絡先: 小出誠二，国立情報学研究所総研大，101-8430 東京都千代田区一ツ橋 2-1-2，koide@nii.ac.jp

Hoare [Hoare 69] は正当性論理式 (Hoare correctness formular)$\{P\}C\{Q\}$ を導入して，プログラム実行を公理的に定義し，プログラムの部分正当性 (停止性を考慮しない) がプログラムに関する構造的帰納法によって合成的に証明できることを示した．ここで $\{P\}$ を事前条件 (*precondition*)，$\{Q\}$ を事後条件 (*precondition*) と呼ぶが，事前条件は STRIPS 様計画問題における前提条件 (*precondition*)，事後条件は効果 (effects) に相当すると捉えることによって，Hoare 論理に基づくプログラム合成問題としてウェブサービス合成問題を捉えなおすことができる．ただし，これまでの多くのプログラム合成研究においてオペレーションは加減乗除と比較演算であり，値域は数や boolean など比較的簡単なものに限られていたが，ウェブサービス合成においてはより複雑に，たとえば OWL において型定義された定数や変数を扱うことになる．したがって，ウェブサービス合成を計算可能にするためには，型解析の機能が必要となる．

前報告 [小出 07, Koide 07] では主にタスク計画の視点からウェブサービス合成問題を議論したが，本報告では，それを踏まえて，自動プログラミングの視点からウェブサービス合成問題を考察する．プログラム自動合成問題は一般に判定不能 (undecidable) であるが，プログラム合成問題からの視点から捉えなおすことにより，ウェブサービス合成とは何であるかをより明確に定義することができると考える．

2. 関数としてのウェブサービス

2.1 ウェブサービスの関数表現

前報告 [小出 07, Koide 07] では階層的タスクネットワーク (HTN) の視点から議論しながら，タスクに関して HTN の視点のみから簡単に議論されていた．本報告では関数としてのタスクという視点からより詳細に議論する．

ウェブサービスの関連タスクをここでは関数として次のように記述する．ここで n は原子タスクの名前シンボル，t は複合タスクの名前シンボルである．

$$\lambda x_i \cdots \lambda x_j.n : X_i \longrightarrow \cdots \longrightarrow X_j \longrightarrow Y$$
$$\lambda x_l \cdots \lambda x_n.t : X_l \longrightarrow \cdots \longrightarrow X_n \longrightarrow Z$$

$X_i \longrightarrow \cdots \longrightarrow X_j \longrightarrow Y$ はこのタスクの型を表している．ここでタスク出力は 1 個とされており，出力変数は明示的に表示されていないが，その型が Y であることだけは指定されている．λx_i はタスク入力 x_i を表すが，その型は関数の型表示において X_i と指定されている．

2.2 簡単なユースケース

本節では簡単なユースケース *BookPriceService*[*1] を通じて，入出力の型に注目したプログラム合成の視点を述べる．今以下のように，三つの原子サービスがあって，エージェントはそれを必要なときに適切に発見できるものとする．

$$\lambda x.BookFinder(x) : BookName \longrightarrow Book$$
$$\lambda x.BNPrice(x) : Book \longrightarrow BookPrice$$
$$\lambda x.\lambda y.CurrencyConverter(x,y) : Price \longrightarrow Currency \longrightarrow Price$$

BookFinder は型 *BookName* のインスタンスを受け取って型 *Book* のインスタンスを返すウェブサービス，*BNPrice* は型 *Book* のインスタンスを受け取って型 *BookPrice* のインスタンスを返すウェブサービス，*CurrencyConverter* は *Price* と *Currency* のインスタンスを受け取って型 *Price* のインスタンスを返すウェブサービスである．ここで，*BookPrice* は *Price* のサブタイプとする．

このユースケースの場合，エージェントはある本の名前と表示させたいカレンシーを入力とし，本の値段 (*BookPrice*) を出力するウェブサービスを入出力の型に着目して簡単に合成することができる．

$$BookPrice = \lambda x.\lambda y.CurrencyConverter(BNPrice(BookPrice(x)), y) : BookName \longrightarrow Currency \longrightarrow BookPrice$$

ただし，上記結果を得るには，型を考慮した単一化計算と型を考慮した融合による証明と同様なシステムが必要である．この例ではウェブサービスの前提条件と効果はないことに注意されたい．前提条件と効果がある場合には，入出力パラメータの型の整合性と同時に，前提条件と効果についても考慮しなければならない．

3. 制御構造

前報告 [小出 07, Koide 07] では階層的タスク計画手法によってウェブサービス合成を行うには，結局制御構造の充足性判定が必要であることを述べ，そのための手法も示した．本報告では，プログラム合成の視点から更に総合的に議論する．

3.1 制御構造の構文

原子タスクは我々の視点ではブラックボックスであるが，複合タスクはその本体に制御構造を含み，制御構造はさらにサブタスクを含んでいる．複合タスクの充足性判定には制御構造の充足性判定が必要である．OWL-S [Martin 04] などではもっと複雑な制御構造が考えられているが，ここでの議論では次のような基本的な制御構造のみを考える．

$$\begin{aligned}
\langle controlConst\rangle &::= \langle task\rangle \mid \langle sequence\rangle \mid \langle conditional\rangle \mid \langle whileLoop\rangle \\
\langle sequence\rangle &::= (\textbf{begin}\ \langle controlConsts\rangle) \\
\langle controlConsts\rangle &::= \emptyset \mid \langle controlConst\rangle \mid \langle controlConst\rangle\ \langle controlConsts\rangle \\
\langle conditional\rangle &::= (\textbf{if}\ \langle condition\rangle\ \langle conseq\rangle\ \langle inconseq\rangle) \\
\langle conseq\rangle &::= \langle controlConst\rangle \\
\langle inconseq\rangle &::= \emptyset \mid \langle controlConst\rangle \\
\langle whileLoop\rangle &::= (\textbf{while}\ \langle condition\rangle\ \langle controlConsts\rangle)
\end{aligned}$$

後の議論では，skip, (begin C_1 C_2), および (while B C) という表記のみ議論するが，skip = (begin), (begin C_1 (begin C_2 $\cdots$)) = (begin C_1 C_2 $\cdots$), (while B C_1 C_2 $\cdots$) = (while B (begin C_1 C_2 $\cdots$)) であることは容易に分かる．

表 1: Hoare 論理における公理

$$\frac{}{\{P\}\ \mathsf{skip}\ \{P\}}\ \text{(skip)}$$

$$\frac{}{\{[\iota \mapsto E]P\}\ \iota := E\ \{P\}}\ \text{(substitution)}$$

$$\frac{\{P_1\}\ C_1\ \{P_2\},\quad \{P_2\}\ C_2\ \{P_3\}}{\{P_1\}(\mathsf{begin}\ C_1\ C_2)\{P_3\}}\ \text{(sequence)}$$

$$\frac{\{P \wedge B\}\ C_1\ \{Q\},\quad \{P \wedge \neg B\}\ C_2\ \{Q\}}{\{P\}(\mathsf{if}\ B\ C_1\ C_2)\{Q\}}\ \text{(conditional)}$$

$$\frac{\{P \wedge B\}\ C\ \{P\}}{\{P\}(\mathsf{while}\ B\ C)\{P \wedge \neg B\}}\ \text{(iteration)}$$

$$\frac{\{P \Rightarrow P'\},\quad \{P'\}\ C\ \{Q'\},\quad \{Q' \Rightarrow Q\}}{\{P\}C\{Q\}}\ \text{(implication)}$$

3.2 制御構造制約

制御構造の充足性について議論するために，**表 1** に示すような Hoare 論理 [Hoare 69] における公理を用いる．表 1 は融合法などの証明のために用いられるルールを表しているが，事前条件 $\{P\}$ や事後条件 $\{Q\}$ は論理式であり，一方 C はプログラムの命令であって両者には意味論的な違いがある．Cousot [Cousot 90] は Hoare 論理について包括的なテキストを書いているが，プログラム言語と論理言語を共通に議論するために，ここでは Cousot に従って，Hoare 論理の意味を議論する．

3.2.1 Hoare 論理の構文

定義(記号): プログラム変数 x, y の集合 **Pvar**，論理変数 X, Y の集合 **Lvar**，変数 ι, ν の集合 **Var** = **Pvar**∪**Lvar**，命令 C の集合 **Com**，定数記号 c の集合 **Cte**，関数記号 f の集合 **Fun**，関係記号 r の集合 **Rel** とする．
定義(項数): $\sharp$ を関数記号と述語記号の項数とする．
定義(論理式構文): 項 T の集合 **Ter**，$T ::= \nu \mid c \mid f(T_1, \ldots, T_{\sharp f})$;
原子論理式 A の集合 **Afo**，$A ::= (T_1 = T_2) \mid (T_1, \ldots, T_{\sharp r})$;
述語論理式 P, Q の集合 **Pre**,

$$P ::= A \mid \neg P \mid (P_1 \wedge P_2) \mid (P_1 \vee P_2) \mid (P_1 \Rightarrow P_2) \mid \exists X.P \mid \forall Y.P$$

Hoare 正当性論理式 H とその集合 **Hef**，$H ::= \{P\}C\{Q\}$, および Hoare 論理式 F とその集合 **HL**，$F ::= P \mid H$ である．
定義(式の構文): 式 E とその集合 **Expr**,

$$E ::= x \mid c \mid f(E_1, \ldots, E_{\sharp f})$$

Boole 式 B とその集合 **BExpr**,

$$B ::= (E_1 = E_2) \mid r(E_1, \ldots, E_{\sharp r}) \mid (\neg B \mid (B_1 \wedge B_2) \mid (B_1 \vee B_2) \mid B_1 \Rightarrow B_2)$$

*1 この例は http://www.mindswap.org/2004/owl-s/1.1/BookPrice.owl から採られた．

表 2: Hoare の証明体系 $\mathbf{H}'$

$\dfrac{}{\{P\}\ \mathsf{skip}\ \{P\}}$	(skip)
$\dfrac{}{\{[\iota \mapsto E]P\}\ \iota := E\ \{P\}}$	(substitution)
$\dfrac{\{P_1\}\ C_1\ \{P_2\},\quad \{P_2\}\ C_2\ \{P_3\}}{\{P_1\}(\mathsf{begin}\ C_1\ \{P_2\}C_2)\{P_3\}}$	(sequence)
$\dfrac{\{P \wedge B\}\ C_1\ \{Q\},\quad \{P \wedge \neg B\}\ C_2\ \{Q\}}{\{P\}(\mathsf{if}\ B\ \{P \wedge B\}C_1\ \{P \wedge \neg B\}C_2)\{Q\}}$	(conditional)
$\dfrac{\{P \wedge B\}\ C\ \{P\}}{\{P\}(\mathsf{while}\ B\ \{P \wedge B\}C\{P\})\{P \wedge \neg B\}}$	(iteration)
$\dfrac{\{P \Rightarrow P'\},\quad \{P'\}\ C\ \{Q\}}{\{P\}\{P'\}C\{Q\}}$	(implication 1)
$\dfrac{\{P\}\ C\ \{Q'\},\quad \{Q'\} \Rightarrow \{Q\}}{\{P\}C\{Q'\}\{Q\}}$	(implication 2)

プログラム言語 **Com** の公理的意味論を定義するために残っていることは，プログラムについて真であると主張できるものすべてを正確に定義することである．ここで意味的には表 1 と同義であるが，より用いることが容易な Hoare の証明体系を**表 2** に与え，プログラム言語 **Com** の表明を以下に与える．

定義(プログラム言語の表明):

$$\begin{aligned} C \quad ::= \quad & \{P_1\}\mathsf{skip}\{P_2\} \mid \{P_1\}\ \iota := E\ \{P_2\} \mid \\ & \{P_1\}(\mathsf{begin}\ C_1\ \{P_2\}C_2)\{P_3\} \mid \\ & \{P_1\}(\mathsf{if}\ B\ \{P_2\}C_1\ \{P_3\}C_2)\{P_4\} \mid \\ & \{P_1\}(\mathsf{while}\ B\ \{P_2\}C\{P_3\})\{P_4\} \mid \\ & \{P\}C \mid C\{P\} \end{aligned}$$

3.2.2 状態と変数代入

さて，変数の代入 $[\iota \mapsto E]P$ についてこれまで何も議論してこなかった．前述のように前提条件や効果に現れる論理変数が，ウェブサービスの入出力にあったとき，これを共通に扱うことのできる枠組みが意味論上の議論においても，実装上の取り扱いにおいても必要である．表 2 を前提にプリミティブな変数代入の形式的な規則は Cousot [Cousot 90] にあるが，ウェブサービス合成では変数は型を有しているためさらに型推論のような詳細な議論が必要となる．ここでは形式化の詳細な定義については述べずに，その変数の代入とそれを扱うための枠組みについて，アイデアの概略を述べる．

Cousot は，与えられた変数の集合 **Var** を定義域とし，空でない計算データ領域 D を値域とする関数 s を考え，これを状態と呼んでいる．命令 C の実行により，それによって引き起こされる代入によって変数の値は変化するが，Cousot はこれを関数 s の変化と捉え，計算途中の状態 s_i とその時点で計算されるべく残されている命令 C_i の組 $\langle s_i, C_i\rangle$ を計算状況 γ_i と定義する．

状態 s における式 $E \in \mathbf{Expr}$ の解釈を $E^{\mathcal{I}} = \mathcal{I}[\![E]\!]$ と表記する．これは $E^{\mathcal{I}}(s) = \mathcal{I}[\![E]\!](s) \in D$ となるような関数を引数とする関数である．すなわち，$E^{\mathcal{I}} = \mathcal{I}[\![E]\!]$ の意味は次のとおり．

$$\mathcal{I} : \mathbf{Expr} \to (S \to D)$$

実装上は状態 s とは変数 x からその値を引き出せるような機構であろう．

3.2.3 Hoare 論理の意味論

Hoare 論理においては前述の状態はプログラム実行中のプログラム変数に割り当てられた値のみならず，事前条件や事後条件における述語論理式で自由変数に対する値を指定するためにも用いられる．Cousot はこれはより複雑なプログラム言語では常に可能とは限らないと述べているが，その詳細は明らかでない．今ここではプログラム変数と論理変数を同じように扱うことができるとする．

項 T の解釈 $T^{\mathcal{I}} = \mathcal{I}[\![T]\!]$，述語論理式 P の解釈 $P^{\mathcal{I}} = \mathcal{I}[\![P]\!]$，および正当性論理式 $\{P\}C\{Q\}$ の意味 $\{\{P\}C\{Q\}\}^{\mathcal{I}} = \mathcal{I}[\![\{P\}C\{Q\}]\!]$ は次のように定義される．

定義(項の解釈):

$$\begin{gathered} \mathcal{I} : \mathbf{Ter} \to (S \to D) \\ \mathcal{I}[\![\nu]\!](s) = s(\nu), \quad \mathcal{I}[\![c]\!](s) = \mathcal{V}[\![c]\!], \\ \mathcal{I}[\![f(T_1, \ldots, T_{\sharp f})]\!](s) = \mathcal{V}[\![f]\!](\mathcal{I}[\![T_1]\!], \ldots, \mathcal{I}[\![T_{\sharp f}]\!]) \end{gathered}$$

ただし，ここで $\mathcal{V}[\![c]\!]$ および $\mathcal{V}[\![f]\!]$ は，それぞれ基本定数，および関数の解釈であり，Cousot はこれらをパラメータとして残しておくことで様々な Hoare 論理の族を定義できると述べている．

定義(述語論理式の解釈):

$$\begin{gathered} \mathcal{I} : \mathbf{Pre} \to \mathbf{Ass} \\ \mathcal{I}[\![T_1 = T_2]\!] = \{s \in S : \langle \mathcal{I}[\![T_1]\!](s), \mathcal{I}[\![T_2]\!](s)\rangle \in \delta, \\ \mathcal{I}[\![r(T_1, \ldots, T_{\sharp r})]\!] = \{s \in S : \langle \mathcal{I}[\![T_1]\!](s), \ldots, \mathcal{I}[\![T_{\sharp r}]\!](s)\rangle \in \mathcal{V}[\![r]\!]\}, \\ \mathcal{I}[\![\neg P]\!] = S - \mathcal{I}[\![P]\!], \quad \mathcal{I}[\![P_1 \wedge P_2]\!] = \mathcal{I}[\![P_1]\!] \cap \mathcal{I}[\![P_2]\!], \\ \mathcal{I}[\![P_1 \vee P_2]\!] = \mathcal{I}[\![P_1]\!] \cup \mathcal{I}[\![P_2]\!], \\ \mathcal{I}[\![P_1 \Rightarrow P_2]\!] = (S - \mathcal{I}[\![P_1]\!]) \cup \mathcal{I}[\![P_2]\!], \\ \mathcal{I}[\![\exists \nu.P]\!] = \{s \in S : (\{s[\nu \mapsto d] : d \in D\} \cap \mathcal{I}[\![P]\!]) \neq \emptyset\}, \\ \mathcal{I}[\![\forall \nu.P]\!] = \{s \in S : (\{s[\nu \mapsto d] : d \in D\} \subseteq \mathcal{I}[\![P]\!])\} \end{gathered}$$

ここで $\delta = \{\langle e, e\rangle : e \in E\}$ は対角集合と呼ばれる E 上の特殊な 2 項関係である．$\mathcal{V}[\![r]\!]$ は関係 r の解釈である．

論理式 $\{P\}C\{Q\}$ の部分正当性について論じる．表明とは状態の集合である．仕様とは状態の表明の対であり，対の左にある要素を入力仕様または事前条件と呼び，右にある要素を出力仕様または事後条件と呼ぶ．命令 C は仕様 $\langle p, q\rangle$ に対して，もし p を満たす初期状態 s で始めた C の任意の終了する実行が q を満たす最終状態 s' で終了しなければならないのなら，これを部分正当であるという．これを $(p \hookleftarrow C^{\mathcal{I}}) \subseteq (S \times q)$ と表現する．$\hookleftarrow$ は 2 項関係 $r \in E \times E$ と E の部分集合 $p \subseteq E$ としたとき，$p \hookleftarrow r = \{\langle e, e'\rangle \in r : e \in p\}$ とする，左制限と呼ばれるものである．

定義(正当性論理式の解釈):

$$\begin{gathered} \mathcal{I} : \mathbf{Hef} \to \{tt, ff\} \\ \mathcal{I}[\![\{P\}C\{Q\}]\!] = \{\mathcal{I}[\![P]\!]\}^{\mathcal{I}} C^{\mathcal{I}} \{\mathcal{I}[\![Q]\!]\}^{\mathcal{I}} \text{ただし} \\ \{p\}^{\mathcal{I}} C^{\mathcal{I}} \{q\}^{\mathcal{I}} = (p \hookleftarrow C^{\mathcal{I}}) \subseteq (S \times q) \end{gathered}$$

ここで tt は Boolean の真，ff は偽を表している．操作的意味論で言い換えると，ある状態 s において

$$\{\mathcal{I}[\![P]\!]\}^{\mathcal{I}} C^{\mathcal{I}} \{\mathcal{I}[\![Q]\!]\}^{\mathcal{I}}(s) = \begin{cases} tt & \text{もし } \mathcal{I}[\![\{P\}]\!](s) = tt \text{ かつ} \\ & C \text{ の終了状態 } s' \text{ において} \\ & \mathcal{I}[\![\{Q\}]\!](s') = tt \\ ff & \text{さもなければ} \end{cases}$$

定義(Hoare 論理式の解釈):

$$\mathcal{I} : \mathbf{HL} \rightarrow \{tt, ff\}$$
$$\mathcal{I}[\![F]\!] = (\text{もし } F \in \mathbf{Pre} \text{ ならば } \mathcal{I}[\![F]\!] = S \text{ さもなければ } \mathcal{I}[\![F]\!])$$

すなわち，Hoare 論理式 **HL** は，述語論理式 **Pre** か正当性論理式 **Hef** であり，前者の場合は $\mathcal{I}[\![F]\!] = S$，後者の場合は $\mathcal{I}[\![F]\!] \in \{tt, ff\}$ である．

3.2.4 状態空間上の論理変数とプログラム変数の統一

前報告 [Koide 07] では階層型タスク計画問題としてウェブサービス問題を捉えたが，そこでは状態空間が用いられ，STRIPS 風な AddList，DelList によって前向きに状態空間は変更された．この状態空間の初期状態には簡単のために，オブジェクトとしての個物とそれらを用いた論理関係式しかないとしている．状態空間における論理変数は型付き単一化 [小出 08] による演算を受けるが，この単一化のための単一化代入子は推論途中において論理変数に対するその時々の値を取り出す機構そのものである．

一方，Lisp や Scheme といったプログラム言語では解釈実行器は次のようなものである．

$$interpet = \lambda e.eval(e, \rho)$$

関数 $eval$ は評価する式 e と並んで自由変数に対する環境 ρ を引数として受け取る．e はユーザが与えるが，ρ はシステムによって保守され，システムは自由変数の値が必要なときには，環境に書かれている情報から自由変数の値を得る．自由変数に値を代入した場合にはその情報は環境 ρ に反映される．我々はウェブサービスを解釈実行するための処理系として Scheme のような言語を提案している．階層型タスクネットワークによる計画において，状態空間とは別に単一化代入子とこの環境 ρ を融合させ，一体化することで，プログラム変数と論理変数の区別をシステム上無くすことができる．

4. おわりに

セマンティック・ウェブサービス合成問題では，これまでタスクプラニングの一種として状態空間中の論理変数とウェブサービスの入出力との相互作用を考慮しなかったり，自動プログラミングの一種として前提条件や効果を無視してきた．本報告では，Hoare 論理の意味論についての Cousot の記述を拡張して，前提条件 (事前条件) 及び効果 (事後条件) と同時に入出力も取り扱うことの意味論を検討し，その実現の枠組みとして，単一化計算における単一化代入子とプログラム言語の $eval$ 引数の環境を統一することを提案した．今後はこのアイデアに従ったシステムを実装する．

前報で述べたように，状態空間中の型を有するオブジェクトと型を有する論理変数との間で単一化計算が行われ，推論が進むたびに解釈はより詳細化されていくが，型推論による計算は入出力変数と環境中の情報とでも行われる．それはタスクを関数として見たとき，一種の型付きラムダ計算と見ることができるが，ウェブサービス合成問題用を型付きラムダ計算として捉えたとき，多くの研究余地が残されている．すなわち，型付きラムダ計算における情報喪失 (information lost) の問題や，型情報の計算に合理的なオントロジー構造の問題があることが分かっているが，さらに研究を深めて次の機会に報告したい．

参考文献

[Ankolekar 04] Ankolekar, A., Paolucci, M., and Sycara, K.: Spinning the OWL-S Process Model Toward the Verification of the OWL-S Process Models, in *Semantic Web Services: Preparing to Meet the World of Business Applications a workshop at 3rd International Semantic Web Conference (ISWC2004)* (2004)

[Cousot 90] Cousot, P.: *Methods and Logics for Proving Programs*, Vol. B: Fromal Models and Semantics of *Handbook of Theoretical Computer Science*, chapter 15, Elsevier (1990), コンピュータ基礎理論ハンドブック II, 形式的モデルと意味論, 細野, 富田 (訳), 819–962, 丸善, 1994.

[Hoare 69] Hoare, C. A. R.: An axiomatic basis for computer programming, *Commun. ACM*, Vol. 12, No. 10, pp. 576–583 (1969)

[Holzmann 03] Holzmann, G. J.: *The SPIN Model Checker: Primer and Reference Manual* (2003)

[Koide 07] Koide, S. and Takeda, H.: Formulation of Hierarchical Task Network Service (De)composition, in *First International Joint Workshop SMR2 2007 on Service Matchmaking and Resource Retrieval in the Semantic Web, Workshop at ISWC2007+ASWC2007*, pp. 107–121 (2007)

[Kuter 05] Kuter, U., Sirin, E., Parsia, B., Nau, D., and Hendler, J.: Information gathering during planning for Web Service composition, *Web Semantics*, Vol. 3, pp. 183–205 (2005)

[Martin 04] Martin, D., Burstein, M., Hobbs, J., McDermott, O. L. D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., and Sycara, N. S. K.: OWL-S: Semantic Markup for Web Services, http://www.w3.org/Submission/2004/ SUBM-OWL-S-20041122/ (2004), W3C

[Nau 03] Nau, D., Au, T.-C., Ilghami, O., Kuter, U., Murdock, J. W., Wu, D., and Yaman, F.: SHOP2: An HTN Planning System, *Journal of Artificial Intelligence Research*, Vol. 20, pp. 379–404 (2003)

[Sirin 04] Sirin, E., Parsia, B., Wu, D., Hendler, J., and Nau, D.: HTN Planning for Web Service Composition Using SHOP2, *Journal of Web Semantics*, Vol. 1, pp. 377–396 (2004)

[Sohrabi 06] Sohrabi, S., Prokoshyna, N., and McIlraith, S. A.: Web Service Composition Via Generic Procedures and Customizing User Preferences, in *The Semantic Web - ISWC 2006*, pp. 597–611, Springer (2006)

[小出 07] 小出 誠二, 武田 英明：階層型計画による Web サービス分解実行, 第 21 回人工知能学会全国大会, pp. 1D1–1, 宮崎 (2007)

[小出 08] 小出 誠二, 武田 英明:OWL による型付き単一化代入, 第 17 回セマンティックウェブとオントロジー研究会 (2008), SIG-SWO-A702-01, http://sigswo.org/papers/SIG-SWO-A702/SIG-SWO-A702-01.pdf