Assume-guarantee Cooperative Satisfaction of Multi-agent Systems

Alireza Partovi[1] and Hai Lin[2]

***Abstract*— This paper aims to investigate the task decomposition problem of multi-agent systems. Task decomposition among agents refers to a process to decompose a given global task into subtasks for individual agents. The decomposition is not arbitrary and should be done in such a way that the satisfaction of the sub-tasks by all agents individually would imply the accomplishment of the global task collectively. In this paper, it is assumed that agents are modeled by labeled transition systems, and the global specification is given as a subclass of Computation Tree Logic (CTL) formulas. It is also assumed that the global CTL specification is broadcasted to and known by all agents. Agents could be heterogeneous and have different capabilities. In order to obtain subtasks for each agent with a maximum potential for fault tolerance, our basic idea is to let each agent contribute to their maximum capabilities in the sense of satisfying a maximum number of sub-formulas of the global specification. The maximum satisfaction set is achieved through a modified CTL model checking algorithm. These maximum satisfiable sub-formulas can be used as the subtask for the corresponding agent. Furthermore, based on assume-guarantee reasoning, sufficient conditions are derived to guarantee the satisfaction of the global CTL specification provided that each agent fullfill its own subtasks. A two-robot cooperative motion planning example is given to illustrate the results.**

I. INTRODUCTION

Multi-agent system (MAS), driven by the technological enhancement in communication, microprocessors, and micro-electro-mechanical systems, has emerged as a hot research area in various applications [1]. The key idea in MAS is to use a group of agents with limited level of autonomy to cooperatively perform a complex task. Multi-agent systems as compared to a single multi-task agent, are potentially more robust due to the possible redundancy in the agents' functionality and also inherently can offer higher adaptability based on the reconfiguration capability.

The key question in MAS is how to design the individual agent's behavior and their coordination rules such that the collective behavior of the team meets the desired global requirements. The vast literature on this topic can be roughly divided into two schools of thought, namely bottom-up and top-down design approaches [2]. The bottom-up design approach is mainly developed to understand how the simple local interaction rules between agents leads to a complicated or coherent global emerging behaviors. Most of the bottom-up design in MAS are heuristic and based on inspirations from natural and social behaviors. It is known that complicated collective behaviors could emerge from very simple local control and coordination among agents. A great deal of efforts have been devoted to the study of collective emerging behaviors under given interaction rules such as the nearest neighbor interaction laws used widely in the consensus based algorithms [3], distributed optimization [4], bottom-up task planing [5], and distributed learning [6]. However, it still remains elusive on how to modify these local control and coordination rules to achieve a certain desire team operations. Consequently, the global specification should be changed accordingly. It is however very difficult to modify the interactions between agents to achieve the new desired collective behaviors. In addition, some emerged collective behaviors could be undesirable, but how to change local interactions to eliminate these undesired emerging behavior is not straightforward in the bottom-up design of MAS. Therefore, the bottom-up design may lead to time consuming trial and error process and hence become inefficient. In order to avoid this inefficiency, top-down design was proposed with the aim to guarantee the correctness of the team behavior by designing the interaction rules and the control laws. The key idea in the top-down design methods is to decompose the global task into the local sub-tasks, such that, if each sub-task as a local specification is fulfilled by an individual agent, the overall emerging behavior meets the global requirements. A critical step in the top-down design is therefore the task decomposition among the agents.

The MAS task decomposition problem was addressed in [7] based on the supervisory control of discrete event systems and the automata theory. In [7], the global specification and individual agents were represented by automata, and the system events were divided to private events set consisting of local sensing and acting actions, and shared events set consisting of synchronous actions between the agents. Each agent extracted the local specification through the natural projection of the global specifications to its own local event set. It was noted that in [7], not all automata could be decomposed such that the parallel composition of obtained sub-task automata is equivalent (in the sense of bisimulation) to the original global automaton. Hence, the authors in [7] investigated the conditions under which such an equivalence holds. The main contribution of [7] is therefore the necessary and sufficient conditions on the global specification automaton which guarantees a successful decomposition. This approach was further evaluated on hieratical formation control of a multi-robot system in [8] and [9]. This framework however, could be fragile with respect to the individual agent, or even single local action failure. In fact, this method requires the global specification to include particulary each individual agent sub-task, and

[1]Alireza Partovi is with Department of Electrical Engineering, Notre Dame University, 252 Fitzpatrick Hall, IN 46556, USA apartovi@nd.edu

[2]Hai Lin is with Department of Electrical Engineering, Notre Dame University, 226A Cushing Hall, IN 46556, USA hlin1@nd.edu

furthermore, expects all the agents perfectly perform their sub-specifications. Hence, a failure in any part of the system potentially could result the team failure. In the follow-up work, they further characterized a class of event failures which the introduced task decomposition framework can tolerate. It was shown in [10] that if the failed event of an agent is passive, i.e., the agent is not the source of that information, the system is still able to decompose the global task. The type of robustness introduced in [10] however, is limited to the communication failure of synchronous actions.

This paper aims to address the robustness issues of task decomposition and propose a new task decomposition scheme of MAS from the compositional model checking and assume-guarantee point of view. Our basic idea is to capitalize the functionality redundancies of the agents. To improve robustness, global tasks do not address the detailed actions that each agents should be responsible. Instead, each agent is allowed to contribute to the team to their maximum potentials. Therefore, if the functionality redundancy in the system permits, failure of an agents could be compensated by the other agents, since each of them individually, tries to maximize its contribution to the global task.

For such a purpose, we use temporal logic formulas to specify the global tasks. Generally speaking, these formulas, by offering operators such as *exist* or *all*, typically do not require a particular chain of behavior to get satisfied. For instant "a robot eventually find the fire", which the task 'finding the fire" is not assigned to a specific agent. In particular, we consider the existential subclass of computation tree logic (CTL). We assumed that the global specification is broadcasted to and known by all individual agents modeled as labeled transition systems. The agents are required to contribute to their maximum capability, and derive the sub-tasks through the maximum achievable sub-formulas of the given CTL formula. The maximum satisfaction is achieved through a modified CTL model checking algorithm [11]. It is worth noting that, due to the distributed nature of the system and also to avoid the state-explosion problem of model checking [12], this algorithm should be applied locally on each agent. The local satisfaction however can be violated by the other agents (environment) when each agent's behavior is evaluated in the team. The individual agents therefore, require their environment to cooperate by respecting a certain behavior, to guarantee their contribution to the global task. We used the assume-guarantee paradigm [13] for this cooperative satisfaction idea. Our main result in this work lies on the derivation of sufficient conditions so to guarantee the satisfaction of the global CTL specification provided that each agent does its best in the sense of satisfying maximum sub-formulas of the global specification.

This work gets inspirations from compositional verification methods [14] used to handel the state-explosion problem of model checking. In these methods a part of the global specification is assigned to each processor, and in order to avoid the state-explosion problem , the model checking is applied to each processor individually. From MAS task decomposition point of view however, these approaches still are fragile with respect to the processor failure as each processor's sub-specification is specifically assigned in the global specification.

The rest of the paper is organized as follows. In Section II the basic concept of labeled transition systems and temporal logic is recalled. Section III introduces the cooperative satisfaction problem and the proposed solution. In Section IV an example is given to illustrate the task decomposition idea in the proposed framework. Finally the paper concludes with remarks in Section V.

II. Preliminaries and definitions

Let's recall some necessary preliminary definitions from [15] as followings.

Definition 1 (Transition Systems): A transition system T is a tuple $T = (S, Act, \rightarrow, I, AP, L)$, where S is a set of state, Act is a set of events, $\rightarrow \subseteq S \times Act \times S$ is a transition relation, $I \subseteq S$ is set of initial states, AP is a atomic propositions set, and $L : S \rightarrow 2^{AP}$ is a labeling function.

The transition system is finite if S, Act, and AP are finite set. In this work, we focus on the transition system with single initial state. It is also assumed the transition system is finite and the transition relation $\rightarrow$ is total, i.e., all the states has at least one outgoing transition, which means the transition system only has infinite path fragments. For simplicity in this paper, we write $s_1 \xrightarrow{a} s_2$ instead of $(s_1, a, s_2) \in \rightarrow$ for $a \in Act$, and use the term *path* instead of *infinite path* from now on.

Definition 2 (Path of Transition System): A *path* in transition system is an infinite state sequence, denoted as $\pi = s_0 s_1 s_2 \cdots$, where $(s_j, s_{j+1}) \in \rightarrow$, $j \geq 0$, which starts from initial states. Let's also denote the set of all pathes in transition system T as $\Pi(T)$.

The focus of this paper is on the asynchronous transition systems which does not introduce any joint actions or shared information. In order to model such systems with mutually exclusive set of events, interleaving operator can be used.

Definition 3 (Interleaving of Transition Systems): Given two transition systems $T_i = (S_i, Act_i, \rightarrow_i, I_i, AP_i, L_i)$, $i = 1, 2$, the interleaving of them is defined as:

$T_1 \,|||\, T_2 = (S_1 \times S_2, Act_1 \cup Act_2, \rightarrow, I_1 \times I_2, AP_1 \cup AP_2, L)$, where the transition relation is

$$\rightarrow \begin{cases} (s_1, s_2) \xrightarrow{\alpha} (s_1', s_2) \text{ if } s_1 \xrightarrow{\alpha}_1 s_1' \\ (s_1, s_2) \xrightarrow{\alpha} (s_1, s_2') \text{ if } s_2 \xrightarrow{\alpha}_2 s_2' \end{cases},$$

and the labeling function L is defined as $L((s_1, s_2)) = L(s_1) \cup L(s_2)$.

The computation tree logic, is a time-branching temporal logic used for specification and verification of hardware and software systems [15]. The time-branching notation of CTL is referred to the fact that, the transition system at each state could have a different possible choice of future decision. The transition system in this perspective, can be unfolded and viewed as a directed tree which has a root of initial states and its branch structured with the states transition. Therefore, a traversal from the tree roots represents a system path, and hence the whole tree interprets all the transition system pathes [15].

The CTL formula over a transition system is interpreted as state and path formulas, which respectively expresses the property of a state and a path. This temporal logic supports an existential path quantifier ($\exists$, exists path) and a universal path quantifier ($\forall$, for all pathes). In this work, our focus is on a sub-class of CTL, which only has existential path quantifier.

Definition 4 (Existential Fragment of CTL): Given a state formula Φ, and a path formula φ over atomic proposition set AP, and let $a \in AP$, the syntax of existential sub-class of CTL ($\exists$CTL) is defined as:

$$\Phi ::= true \mid false \mid a \mid \neg a \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid \exists \varphi$$
$$\varphi ::= \bigcirc \Phi \mid \Phi_1 \sqcup \Phi_2 \mid \Phi_1 \,\mathrm{R}\, \Phi_2.$$

In the introduced ($\exists$CTL) syntax, the formula $\bigcirc\Phi$ is true if the next state in the path holds Φ, the formula used until operator, $\Phi_1 \sqcup \Phi_2$, is true if Φ_1 is held align the path prior to a state that satisfies Φ_2, and operator release in $\Phi_1 \,\mathrm{R}\, \Phi_2$ is translated as either Φ_1 is true before $\neg\Phi_2$ holds, or Φ_2 holds globally along the path.

Verification of a CTL formula over a transition system is defined as a satisfaction relation. Formally, satisfaction of a state formula ϕ over a state in transition system, is defined as $s \models \phi$, which is true if and only if ϕ holds on state s. Similarly for a path formula φ, and a path π in transition system, we have $\pi \models \varphi$ which is true if and only if φ holds along the path π.

Definition 5 (Satisfaction Relation for $\exists$CTL): Consider a transition system $T = (S, Act, \rightarrow, I, AP, L)$, and let Ψ, Φ_1, Φ_2 be $\exists$CTL states formula, φ a path formula, $a \in AP$ an atomic proposition, π be a transition system path, and $s \in S$. The satisfaction relationship for $\exists$CTL is defined as follows.

$$\begin{array}{lll}
s \models a & \text{iff} & a \in L(s) \\
s \models \neg\Psi & \text{iff} & \text{not } s \models \Psi \\
s \models \Phi_1 \wedge \Phi_2 & \text{iff} & (s \models \Phi_1) \text{ and } (s \models \Phi_2) \\
s \models \Phi_1 \vee \Phi_2 & \text{iff} & (s \models \Phi_1) \text{ or } (s \models \Phi_2) \\
s \models \exists\varphi & \text{iff} & \pi \models \varphi \text{ for some } \pi \in \Pi(s) \\
\pi \models \bigcirc\Psi & \text{iff} & \pi[1] \models \Psi \\
\pi \models \Phi_1 \sqcup \Phi_2 & \text{iff} & \exists j \geq 0, \pi[j] \models \Phi_2 \\
 & & \wedge ((\forall 0 \leq l < j, \pi[l] \models \Phi) \\
\pi \models \Phi_1 \,\mathrm{R}\, \Phi_2 & \text{iff} & \forall j \geq 0, (\pi[j..] \models \Phi_2 \\
 & & \vee \exists i < j, \pi[i..] \models \Phi_2)
\end{array}$$

The CTL semantic for transition system is defined in the Definition 6.

Definition 6 (CTL semantic for Transition System): A satisfaction set of CTL state formula Φ over a transition system T, is denote as $\mathrm{Sat}(\Phi)$, and is defined by $\mathrm{Sat}(\Phi) = \{s \in S \mid s \models \Phi\}$, and as for the transition system, we have $T \models \Phi$ if and only if $I \subseteq \mathrm{Sat}(\Phi)$.

The satisfaction set of a CTL formula can be obtain through CTL model checking which is given in Algorithm 1.

Algorithm 1 (Model Checking Basic Algorithm): This algorithm is the basic idea of CTL model checking addressed in [15]. The only difference is ignoring the sub-formulas which the transition system is not able to satisfy, i.e., $\mathrm{Sat}(\phi_i) = \emptyset$. This basic algorithm provides the maximum sub-formulas of the given CTL formula that the transition system satisfies.
Input: finite transition system T_i, and CTL formula Φ, over atomic proposition AP.
Output: All the Φ sub-formula, that transition T_i satisfies.

```
for all i ≤ |Φ| do
  for all Ψ ∈ Sub(Φ) with |Ψ| = i do
    compute Sat(Ψ) from Sat(Ψ′)
    if Sat(Ψ) = ∅ ignore it.
  loop
loop
return I ⊆ Sat(Ψ)
```

III. Cooperative Satisfaction Problem

The cooperative satisfaction in this paper is introduced in the following perspective. It is assumed that there is a group of interleaved transition system over the same atomic proposition. Each transition system receives the global specification which is given in $\exists$CTL format, and tries its best to satisfy it by providing the maximum sub-formulas it can locally satisfy. This information is obtained by locally performing the model checking algorithm on the given global specification. Then an interesting question is to find out under what conditions, it is possible to conclude on achieving the global specification by having the local satisfaction set. In fact in this framework each transition system contributes to satisfy the global specification to maximum of its capability (cooperative satisfaction). This problem for two transition systems, formally can be defined as Problem 1.

Problem 1: Let the global specification be given in $\exists$CTL formula, denoted as Φ_G, and assume that the distributed system is interleaved of two labeled transition systems $T_i = (S_i, Act_i, \rightarrow_i, I_i, AP, L_i)$, $i = 1, 2$, define as $T_G = |||_{i=1}^{2} T_i$. If each transition system locally satisfies a part of the global specification, denoted as ψ_i, then it is desire to characterize the conditions on ψ_i, which results the composed system satisfies the given global specification, i.e., $T_G \models \Phi_G$.

A. Assume-guarantee Reasoning

Since in this introduced framework, the behavior of each transition system depends on its environment, it may not be possible to conclude on the global property based on the local behavior. Assume-guarantee reasoning provides a paradigm in which a local property can be guaranteed in a global perspective. In this paradigm, the reasoning is based on committing to satisfy a specification (guarantee) by requiring the environment to respect a ceratin behavior (assume). This approach has been used for compositional verification of concurrent systems [16]. The assume-guarantee paradigm is denoted as $<\psi_E> T <\psi>$, where ψ_E is the assumption for the environment of the transition system T, and ψ is the specification that transition system T can guarantee to satisfy. It formally interpreted as $T \models \psi$, if the assumption of ψ_E on the environment T_E is inherently true, i.e., $<True> T_E < \psi_E >$. Using this paradigm, we propose Theorem 1 for the cooperative satisfaction problem of two interleaved transition systems.

Theorem 1: Given interleaved of two transition systems as $T_G = |||_{i=1}^{2} T_i$, environment of T_i as $T_{E_j} = |||_{i=1,i\neq j}^{2} T_i$, and the global specification as conjunctive-disjunctive form denoted as $\Phi_G = \bigvee_{c=1}^{\bar{c}} \phi_c \wedge \bigwedge_{k=1}^{\bar{k}} \phi_k$, where ϕ_c and ϕ_k are $\exists$CTL sub-formula in form of $\exists\varphi$ defined as in Definition 4.
If $<\psi_{E_i}><T_i><\psi_i>$ with $<True><T_{E_i}><\psi_i>$ holds on each transition system, where $i = \{1,2\}$, $\psi_i = \Phi_{Ci} \wedge \Phi_{Ki}$, $\Phi_C = \bigvee_{c\in C} \phi_c$, $C \subseteq \{1 \cdots \bar{c}\}$, $\Phi_K = \bigwedge_{k\in K} \phi_k$, and ψ_{E_i} is inductively obtained from Table I. The composed system T_G satisfies the global specification, $T_G \models \Phi_G$, if $(\bigvee_{i=1}^{2} \Phi_{Ci} \wedge \bigwedge_{i=1}^{2} \Phi_{Ki}) \rightarrow \Phi_G$.

TABLE I

ENVIRONMENT ASSUMPTION TABLE FOR INTERLEAVED SYSTEMS.

ψ	*Not Violated*, $\overline{\nabla}(\psi)$	*Assumption*, ψ_E
a	True	True
$\neg a$	$\neg a$	$\neg a$
$\Phi_1 \wedge \Phi_2$	$\overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_2)$	$\overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_2)$
$\Phi_1 \vee \Phi_2$	$\overline{\nabla}(\Phi_1) \vee \overline{\nabla}(\Phi_2)$	$\overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_2)$ if $T \models \Phi_1 \wedge \Phi_2$ $\overline{\nabla}(\Phi_1)$ if $T \models \Phi_1$ $\overline{\nabla}(\Phi_2)$ if $T \models \Phi_2$
$\exists \bigcirc \Phi$	$\overline{\nabla}(\Phi)$	$\overline{\nabla}(\Phi)$
$\exists \Phi_1 \sqcup \Phi_2$	$\overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_2)$	$[\overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_2)]$ $\vee [\exists \Diamond \overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_2)]$ if $l > 0$. $[\overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_2)] \vee \overline{\nabla}(\Phi_2)$ if $l = 0$.
$\exists \Phi_1 \mathrm{R} \Phi_2$	$\overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_2)$	$\overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_2)$

In Table I, l is a the counter variable which is defined in Definition 5 for until operator. The column *Not Violated*, is actually the most conservative assumption on behavior of the environment which if it is respected, the local transition system behavior is not violated in the composed system. The $\overline{\nabla}(.)$ operator can be obtained inductively form the transition system specification, denoted as ψ in Table I. The column *assumption* is the direct consequences of *Not Violated* section results in a less conservative sense which may need the information on how the local transition system satisfies ψ.

Proof: [Theorem 1] Let's assume that the transition system $T \models \psi$, where $\psi = \Phi_C \wedge \Phi_K$, and therefore there must be a path $\pi_\psi \in \Pi(T)$ such that $\pi_\psi \models \psi$. Let's denote Π_ψ the set of all pathes in T that satisfies ψ, $\Pi_\psi = \{\forall \pi \in \Pi(T) \mid \pi \models \psi\}$. The first aim is to find an assumption on the environment of the transition system T_E, such that $<\psi_E><T><\psi>$ implies $T \,|||\, T_E \models \psi$. This part of proof is based on the induction hypothesis.

Induction basis: Let $T = (S, Act_1, \rightarrow_1, s_0, AP, L_1)$, $T_E = (Q, Act_2, \rightarrow_2, q_0, AP, L_2)$, $a \in AP$ be an atomic proposition, $T_G = T \,|||\, T_E$, the initial state of composed system denoted as $I = (s_0, q_0)$, and Φ, Φ_1, Φ_2 be the $\exists$CTL formulas.

Induction steps : Case 1: $\psi = a$. Since $T \models \psi$, then $s_0 \models \psi$ which implies $a \in L_1(s_0)$ and correspondingly $a \in L(I) = L_1(s_0) \cup L_2(q_0)$. Hence, $I \models \psi$, and the composed system also satisfies ψ, i.e., $T_G \models \psi$. Therefore, no assumption on the environment is required for this case, i.e., $<True> T <\psi>$ implies $T_G \models \psi$. As a result $\psi_E = True$.

Case 2: $\psi = \neg a$. Since $T \models \psi$, then $s_0 \models \psi$, which implies $a \notin L_1(s_0)$. In this case $I \models \psi$, provided that $a \notin L(I) = L_1(s_0) \cup L_2(q0)$, which requires that $a \notin L_2(q_0)$. Hence, if $T_E \models \neg a$, the composed system also satisfies ψ, therefore $\overline{\nabla}(\psi) = \neg a$, and $\psi_E = \neg a$.

Case 3: $\psi = \Phi_1 \wedge \Phi_2$. Since $T \models \psi$, then $T \models \Phi_1$ and $T \models \Phi_2$. Therefore, $\overline{\nabla}(\psi) = \overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_1)$, and $\psi_E = \overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_1)$.

Case 4: $\psi = \Phi_1 \vee \Phi_2$. Since $T \models \psi$, then $T \models \Phi_1$ or $T \models \Phi_2$. In terms of violation, we consider the conservative case that the environment should not violates the whole ψ, $\overline{\nabla}(\psi) = \overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_1)$. However, if either of them makes the composed system to satisfy Φ_1 or Φ_2 is sufficient. Hence, it is possible to have different environment assumptions for different possibilities : $\psi_E = \overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_2)$ if $T \models \Phi_1 \wedge \Phi_2$, $\psi_E = \overline{\nabla}(\Phi_1)$ if $T \models \Phi_1$, and $\psi_E = \overline{\nabla}(\Phi_2)$ if $T \models \Phi_2$.

Case 5: $\psi = \exists \bigcirc \Phi$. Since $T \models \psi$, then there exists $\pi_\psi \in \Pi_\psi$, such that $\pi_\psi[1] \models \Phi$. Consider a path $\lambda \in \Pi(T_G)$ which has the fragment of $\lambda = (\pi_\psi[0], q_0)(\pi_\psi[1], q_0) \cdots (\pi_\psi[\infty], q_0)$. It is clear that if $(\pi_\psi[1], q_0) \models \Phi$, then $\lambda \models \psi$ and correspondingly $T_G \models \psi$, which hence requires that $q_0 \models \overline{\nabla}(\Phi)$. As a result $\overline{\nabla}(\psi) = \overline{\nabla}(\Phi)$, and $\psi_E = \overline{\nabla}(\Phi)$.

Case 6: $\psi = \exists \Phi_1 \sqcup \Phi_2$. Since $T \models \psi$, then there exists $k \geq 0$ and $0 \leq i \leq k$ such that $(\pi_\psi[k] \models \Phi_2) \wedge (\pi_\psi[i] \models \Phi_1)$. Let's consider the following two possibilities : 1) $T \models \psi$ with $k = 0$, which implies $s_0 \models \Phi_2$. In this case as long as $q_0 \models \overline{\nabla}(\Phi_2)$, it can be conclude that $(s_0, q_0) \models \Phi_2$ which yields $T_G \models \psi$. 2) $T \models \psi$ with $k > 0$, for this case let's consider a path $\lambda \in \Pi(T_G)$ which has the fragment of $\lambda = (\pi_\psi[0], q_0) \cdots (\pi_\psi[0], q)(\pi_\psi[1], q) \cdots (\pi_\psi[\infty], q)$, where $q \in Q$. In this path, if the state q does not violate $\pi_\psi \models \Phi_1 \wedge \Phi_2$, then $\lambda \models \psi$, which results $T_G \models \psi$. In terms of violation, we consider the conservative case, that the environment should not violates the whole ψ, $\overline{\nabla}(\psi) = \overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_1)$. However in terms of assumption, we can have the following cases: $\psi_E = [\overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_2)] \vee \exists\Diamond[\overline{\nabla}(\Phi_1) \wedge \overline{\nabla}(\Phi_2)]$ if $k > 0$, and $\psi_E = \overline{\nabla}(\Phi_2)$ if $k = 0$.

Case 7: $\psi = \exists \Phi_1 \,\mathrm{R}\, \Phi_2$. Similar reasoning of Case 5 can be applied for this semantic.

Thus, by induction hypothesis, for any $\exists$CTL specification that the transition system locally satisfies, $T \models \psi$, it is possible to obtain the environment assumption which if it holds, $T_G \models \psi$ can be guaranteed. Formally, given $< \psi_{E_i} > T_i < \psi_i >$, where ψ_{E_i} is obtained from Table I, if $< True > T_{E_i} < \psi_{E_i} >$, it implies $< True > T_G < \psi_i >$.

Furthermore, if $< True > T_G < \psi_i >$ holds for any ψ_i, then provided that the combination of ψ_i, implies the global specification formula, it can be concluded that the composed system is able to satisfy the given global specification. Formally, for any $\psi_i = \Phi_{Ci} \wedge \Phi_{Ki}$ which $< True > T_{E_i} < \psi_i >$ holds, if $(\bigvee_{i=1}^{2} \Phi_{Ci} \wedge \bigwedge_{i=1}^{2} \Phi_{Ki}) \rightarrow \Phi_G$, then $T_G \models \Phi_G$. The proof is complete. ∎

Remark 1: From the syntax of $\exists$CTL in Definition 4, it can be observed that every $\exists$CTL can always be written in disjunctive-conjunctive from.

Remark 2: In the problem formulation, the local satisfaction set ψ_i is actually all the sub-formulas of global specification Φ_G, that the transition system T_i, is able to locally satisfy. In order to provide ψ_i, it is assume that each transition system does the model checking on Φ_G based on algorithm given in Algorithm 1.

The MAS task decomposition through the cooperative satisfaction perspective can be explained as follows. Each agent receives the team task (global specification) which is given as a temporal logic formula. Through the local model checking, it obtains the sub-formula set that it can locally satisfy. The sub-formula set, depending on the application, can be interpreted as all the sub-tasks that each agent is capable to achieve. The achievement however is hold provided that the agent's environment cooperates by respecting a ceratin behavior, defined as the environment assumption. In the proposed framework, the environment assumption is supposed to be the property of the environment which inherently holds. As a result, if the combination of all local sub-formulas which the corresponding assumption is held on the environment, implies the global specification, it be can be concluded that the overall system is capable to achieve the team task.

IV. Example

The following example is used to illustrate the application of cooperative satisfaction for motion planning of multi-robot system.

Example 1: Consider the map depicted in Figure 4 which shows a partitioned region with unique section's label. Let's assume each robot only has access to some sections. The specification in this example can be defined as asking robots to reach a certain section while avoiding the prohibited regions. The main goal here is to check if there is a possible solution (robot move), such that the requested collective robots motion can be achieved. Let's further assume that the robots functionality is modeled as the transition systems depicted in Figures 1 and 2. The label of each accessible section can be considered as an atomic proposition, and therefore the atomic proposition set can be define as $AP = \{\pi_1, \pi_2, \pi_3, \pi_4, \pi_5, \pi_6, \pi_7, \pi_8, \pi_9\}$. The action set for this example is not important to be specifically defined, however it can be interpreted as some robot motion actions such as $\{$move left, move right, move up, move down$\}$. The initial state, and labeling function also are clear from the transition system models in Figures 1 and 2.

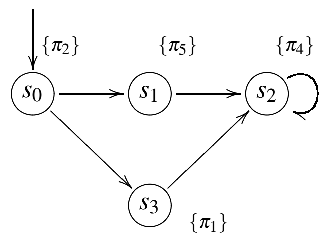

Fig. 1. Labeled transition system of Robot1.

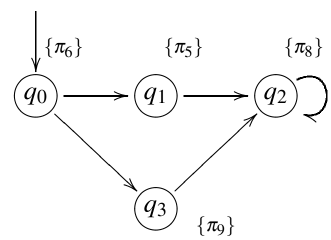

Fig. 2. Labeled transition system of Robot2.

Let's assume the desire motion plan for this group of robot is :

1) At least one robot has to reach section π_4.
2) At least one robot has to reach section π_8.
3) No robot pass over π_5.

This specification can be realized with the following $\exists$CTL formula, $\Phi_G = \exists\Diamond\pi_4 \wedge \exists\Diamond\pi_8 \wedge \exists\Box\neg\pi_5$ which in a compact conjunctive form is $\Phi_G = \bigwedge_{i=1}^{2} \phi_i$, where $\phi_1 = \exists\Diamond\pi_4$, $\phi_2 = \exists\Diamond\pi_8$, and $\phi_3 = \exists\Box\neg\pi_5$. Each robot receives the global specification and performs a model checking on their own transition system. Robot1 is only able to reach region π_4 and Robot2 only can go to region π_8, while both are able to avoid π_5. The satisfied sub-formulas for each robot trivially is : $T_1 \models \phi_1 \wedge \phi_3 := \psi_1$, and $T_2 \models \phi_2 \wedge \phi_3 := \psi_2$. Hence, by converting the operator *evetually* $\Diamond$ and *always* $\Box$ to the basic operators defined in Defenition 4 [15], and using Table I, the environment assumption for T_1 can be obtained as

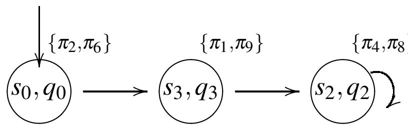

Fig. 3. One of the possible solution of robot motion for Example 1.

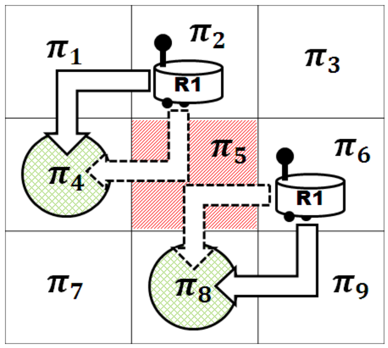

Fig. 4. The robots' motion map with illustration of a possible robots path that solves Example 1.

$\psi_{E_1} = \text{True} \wedge \neg\pi_5 = \neg\pi_5$ which holds on $T_{E_1} = T_2$. Similarly for second transition system T_2, $\psi_{E_2} = \text{True} \wedge \neg\pi_5 = \neg\pi_5$, which also holds on $T_{E_2} = T_1$. Furthermore, since $\psi_1 \wedge \psi_2 \rightarrow \Phi_G$, the composed system satisfies the global specification, i.e., $|||_{i=1}^{2} T_i \models \Phi_G$. This conclusion can be verified by constructing the interleaved system and verify Φ_G on it. However, trivially by looking on the possible robots motion path, it can be observed that, event though no robot individually can reach to the section π_4 and π_8, since they move independently, each robot is able to reach to the destinations while both can avoid the forbidden section π_5. Verification on the composed system shows there are many possible robots motion that solve this problem, Figure 3 gives one of this possible chain movement.

V. Conclusion

This work proposed a new approach for task decomposition of multi-agent systems through the scheme of temporal logic and assume-guarantee paradigm which can potentially enhance the system robustness. In the introduced method, the global specification is given by temporal logic formula which does not define specifically each agent's task. The individual agents in fact are required to contribute to the team's task up to their maximum capability by providing all the locally achievable sub-formulas contains in the global specification. Using the assume-guarantee reasoning, each agent guarantees that the locally satisfied sub-formulas holds globally by requesting its environment to respect a certain systematically driven assumption. It is shown that, providing the environments hold the requested property, if the satisfaction set of each individual agent collectively implies the global specification, the composed system is able to satisfy the desired group requirement.

VI. Acknowledgments

The financial supports from NSF-CNS-1239222 and NSF-EECS-1253488 for this work are greatly acknowledged.

References

[1] V. Lesser, "Cooperative multiagent systems: a personal view of the state of the art," *IEEE Transactions on Knowledge and Data Engineering*, vol. 11, pp. 133 –142, Feb. 1999.

[2] V. Crespi, A. Galstyan, and K. Lerman, "Top-down vs bottom-up methodologies in multi-agent system design," *Autonomous Robots*, vol. 24, no. 3, pp. 303–313, 2008.

[3] W. Ren, R. W. Beard, and E. M. Atkins, "A survey of consensus problems in multi-agent coordination," in *American Control Conference, 2005. Proceedings of the 2005*. IEEE, 2005, pp. 1859–1864.

[4] R. L. Raffard, C. J. Tomlin, and S. P. Boyd, "Distributed optimization for cooperative agents: Application to formation flight," in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 3. IEEE, 2004, pp. 2453–2459.

[5] E. H. Durfee, "Distributed problem solving and planning," in *Multi-agent systems and applications*. Springer, 2001, pp. 118–149.

[6] J. Choi, S. Oh, and R. Horowitz, "Distributed learning and cooperative control for multi-agent systems," *Automatica*, vol. 45, no. 12, pp. 2802–2814, 2009.

[7] M. Karimadini, "Cooperative tasking for multi-agent systems," Ph.D. dissertation, National University of Singapore, 2011.

[8] A. Karimoddini, H. Lin, B. M. Chen, and T. H. Lee, "Hybrid three-dimensional formation control for unmanned helicopters," *Automatica*, vol. 49, no. 2, pp. 424–433, 2013.

[9] A. Karimoddini, M. Karimadini, and L. Hai, "Decentralized hybrid formation control of unmanned aerial vehicles," in *American Control Conference, 2014*.

[10] M. Karimadini and H. Lin, "Fault-tolerant cooperative tasking for multi-agent systems," *International Journal of Control*, vol. 84, no. 12, pp. 2092–2107, 2011.

[11] E. M. Clarke, E. A. Emerson, and A. P. Sistla, "Automatic verification of finite-state concurrent systems using temporal logic specifications," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 8, no. 2, pp. 244–263, 1986.

[12] Clarke, O. Grumberg, and D. A. Peled, *Model checking*. MIT Press, 1999.

[13] "You assume, we guarantee: Methodology and case studies," ser. Lecture Notes in Computer Science, vol. 1427.

[14] H. Peng and S. Tahar, "A survey on compositional verification," Citeseer, Tech. Rep., 1998.

[15] C. Baier, J.-P. Katoen, *et al.*, *Principles of model checking*. MIT press Cambridge, 2008, vol. 26202649.

[16] W.-P. d. Roever, F. d. Boer, U. Hanneman, J. Hooman, Y. Lakhnech, M. Poel, and J. Zwiers, *Concurrency Verification: Introduction to Compositional and Non-compositional Methods*. Cambridge University Press, Nov. 2001.