Generating and Choosing Organizations for Multi-Agent Systems - Extended Abstract

Cleber J. Amaral
Instituto Federal de Santa Catarina
São José, Brazil
cleber.amaral@ifsc.edu.br

Jomi F. Hübner
Universidade Federal de Santa Catarina
Florianópolis, Brazil
jomi.hubner@ufsc.br

Stephen Cranefield
University of Otago
Dunedin, New Zealand
stephen.cranefield@otago.ac.nz

ABSTRACT

The design of organizations is a complex and laborious task. It is the subject of recent studies, which define models to automatically perform this task. However, existing models constrain the space of possible solutions by requiring a priori definitions of organizational roles and usually are not suitable for planning resource use. This paper presents GoOrg [1], a model that uses as input a set of goals and a set of available agents to generate different arrangements of organizational structures made up of synthesized organizational positions. The most distinguishing characteristics of GoOrg are the use of organizational positions instead of roles and that positions are automatically synthesized rather than required as a priori defined inputs. These features allow for the planning of organizational resources at design time and increase the chance of finding feasible solutions. This paper also introduces a model extension that illustrates how GoOrg can be extended to suit a specific domain.

KEYWORDS

Organization design, Organizational structure, Automated design.

ACM Reference Format:

1 INTRODUCTION

Existing models for generating organizational structures for multi-agent systems [2–5, 7] use the concept of roles which has many-to-many relationships with agents and should be defined a priori by the user. GoOrg uses organizational positions instead, which have one-to-one relationships with agents [6], thus an organizational structure made up of positions reflects resource demands. This allows the user to know in advance (at design time) the resources required for an organization to run. In GoOrg, positions are synthesized rather than required as input. It creates a broader range of possible solutions, increasing the possibility of finding structures that the available agents can fill. GoOrg also addresses the challenge of choosing a solution among many possibilities, using quantified characteristics for the problem domain.

2 GOORG MODEL

The GoOrg model considers only essential elements for an organization’s design: goals, agents, organizational positions, features, and the organizational structure. A goal $g \in G \subset \text{symbols}$ is a desired state of the world that the organization hopes to attain. An agent $a \in A \subset \text{symbols}$ is an entity that acts to achieve the goals it is committed to. A position $p \in P \subset \text{symbols}$ is a place-holder for an agent in an organization. Positions represent the agents necessary for an organization to function. The agent who occupies a position is in charge of achieving the goals assigned to that position. The goals assigned to a position $p$ are specified by the function $gp$. The function $ap$ specifies the agent occupying the position $p$, considering that $p$ is a “free position” when $ap(p) = \epsilon$.

$$gp : P \rightarrow 2^G, \forall p \in P, gp(p) \neq \{\}, \text{ap} : P \rightarrow A \cup \{\epsilon\}$$

An agent cannot be bound to more than one position (i.e., $\forall p, p' \in P, (p \neq p') \land (ap(p) \neq \epsilon) \land (ap(p') \neq \epsilon) \Rightarrow (ap(p) \neq ap(p'))$). To check if an agent can occupy a position, GoOrg compares the features that an agent has to the features that the goals assigned to a position have. A feature $f$ is an n-tuple, in which the first element is a symbol. Besides the first element, optionally, a feature may have other elements $(e_2, \ldots, e_n)$. The function $fg$ specifies the features required by a goal. The function $fa$ specifies the features an agent has.

$$f : \langle \text{symbol}, e_2, \ldots, e_n \rangle, f \in F, fg : G \rightarrow 2^F, fa : A \rightarrow 2^F$$

GoOrg considers that each organizational structure is a particular description of an organization. An organizational structure $o$ is represented as a tuple.

$$o : \langle G, A, P, gp, fg, fa, ap \rangle$$

Each generated organization has attributes that quantify it. GoOrg defines the attribute feasibility represented as $\kappa(o)$, a real number in the range $[0,1]$. If every organizational position has an agent to occupy it, the organization is considered feasible ($\kappa(o) = 1$).

$$\kappa(o) = \frac{|\{ap(p) | ap(p) \neq \epsilon, p \in P\}|}{|P|}$$

3 GOORG4PROD: AN EXTENSION OF GOORG

GoOrg must be extended to address the requirements of each domain. GoOrg4Prod generates structures of positions responsible for production activities in a factory. To address it, GoOrg4Prod specifies that organizational goals are associated with workloads,
as efforts that should be performed by skilled agents. Thus, a workload \( w \) represents a demanded effort \( e \in \mathbb{R}^+ \) which requires a skill \( s \in S \subset \text{symbols} \) to be performed.

\[
w : (s, e), w \in W, \ \text{w} : G \rightarrow 2^W
\]

Fig. 1 highlights the organizational attributes and features added by GoOrg4Prod to extend the GoOrg model. GoOrg4Prod matches agents and positions using skills. It uses workloads to calculate the organization’s efficiency, which among other attributes can be used to choose organizations based on the user’s preferences.

**Figure 1: GoOrg4Prod model.**

GoOrg4Prod considers that organizational positions may have superordinate-subordinate relationships, which are represented as “is superior of” relationships. The function \( sp : P \rightarrow P \cup \{e\} \) records the position \( p' \), which is the immediate superordinate of the position \( p \). If \( p \) has “no superordinate”, \( sp(p) = e \). In this extension of GoOrg, an organizational structure is defined as follows:

\[
o : (G, A, P, F, qp, fg, fa, ap, sp, wg)
\]

For generating structures, GoOrg4Prod perform a search in the space, applying three structure transformations: (i) \( \text{addSuperior}(g) \) synthesizes a superordinate position and assigns the goal \( g \) to it; (ii) \( \text{addSubordinate}(g, p') \) synthesizes the position \( p' \) as subordinate of \( p \) and assigns the goal \( g \) to \( p \); and (iii) \( \text{joinPosition}(g, p) \) assigns the goal \( g \) to a previously synthesized position \( p \).

Based on superordinate-subordinate relationships, each structure’s height is calculated. It refers to how centralized and bureaucratic an hierarchical organization is. Based on how the goals are distributed across positions, the generality is calculated. An organization with high generality requires agents with more skills, i.e., more generalist agents, and the opposite requires more specialist agents. With the feature workload, the efficiency of an organization can be quantified. It indicates how close the combined capacity of the agents, who will occupy the synthesized positions, is to the expected efforts considering the given goals. For choosing an organization among the list of solutions, a partial order relation representing the user’s preferences is defined as \( o \preceq o' \) in which \( o \preceq o' \) means that \( o \) is preferred to \( o' \). Each organizational attribute (or its complement) is a criterion \( c \). A priority order is represented by a natural number \( y \in \Gamma \), in which \( c_1 \) is the most important criterion for the user. For instance, if two criteria were set (\( \Gamma = \{1, 2\} \)), \( o \preceq o' \) is defined as:

\[
o \preceq o' \iff [c_1(o) > c_1(o') \lor (c_1(o) = c_1(o') \land c_2(o) > c_2(o'))]
\]

GoOrg4Prod checks the organization’s feasibility by matching the features of the available agents and of the synthesized positions.

**4 RESULTS**

As a motivating scenario, the agents have to access a database to get orders, get boxes from shelves, move them near the conveyor belt, and finally pick items from the boxes to place on the head of the conveyor belt. These activities are specified as the goals FeedProduction, GetBox, MoveBox$0, MoveBox$1 and PlaceBox that are associated with workloads. Some skills are required to achieve these goals: db access, lift, move and pnp (pick and place).

It is assumed that the user prefers the most generalist, efficient and flat structure in this priority order, and there are three kinds of agents available: LE, an agent with the skills lift and database access for lifting boxes on shelves and to program access to an external database; BT, an agent with the skill move for moving boxes around the floor; and PP, an agent with the skill pick and place for picking items from the box and placing them on the conveyor belt.

For this problem, GoOrg4Prod has produced 1,646 organizational structure candidates, each having all the given goals assigned to positions. The left-hand side of Fig. 2 presents the candidate #1, which has only one organizational position. In this structure, one agent alone is responsible for achieving all organizational goals. This solution has 100% of generality since all positions (only one in this case) are assigned to all goals. It also has the minimum height and has the highest efficiency for this problem. Although it is the best candidate according to the user’s preferences, this solution is not feasible since there is no available agent that has all four required skills (db access, lift, move and pnp).

![Figure 2: The best (left) and the first feasible candidate (right).](image)

The right-hand side of Fig. 2 presents the candidate #134, which is another of the flattest candidates (just one hierarchy level). However, it has lower generality compared to the candidate #1 and it has lower efficiency (as it has three positions). Although it is not the ideal solution (candidate #1), taking the given available agents and the user’s preferences, this candidate is GoOrg4Prod’s first choice since it is the first one that is 100% feasible.

**5 CONCLUSION**

GoOrg has only the fundamental elements that are found in any organizational design. It is extensible for dealing with the specificity and complexity of each domain. This study adopted positions instead of roles for designing organizational structures. The reason is that positions carry the same advantage of the roles in respect to being detached from named agents, while numerically reflecting the need for resources. It means that the feasibility for a specific state of an organization can be checked during the design. The generated candidates have quantified attributes which enables a multi-criteria approach to choose the “best” organization.

**ACKNOWLEDGMENTS**

The research was partially funded by Project AG-BR Petrobras and Programme PrInt CAPES-UFSC “Amação 4.0”.

2825
REFERENCES


