2017

Implementation and Evaluation of Goal Selection in a Cognitive Architecture

Sravya Kondrakunta
Wright State University

Follow this and additional works at: https://corescholar.libraries.wright.edu/etd_all

Part of the Computer Engineering Commons, and the Computer Sciences Commons

Repository Citation
https://corescholar.libraries.wright.edu/etd_all/1811

This Thesis is brought to you for free and open access by the Theses and Dissertations at CORE Scholar. It has been accepted for inclusion in Browse all Theses and Dissertations by an authorized administrator of CORE Scholar. For more information, please contact library-corescholar@wright.edu.
Implementation and Evaluation of Goal Selection in a Cognitive Architecture

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science

By

Sravya Kondrakunta
B.Tech., Koneru Lakshmaiah University, 2015

2017
Wright State University

____________________________________
Michelle A. Cheatham, Ph.D.
Thesis Director

____________________________________
Mateen M. Rizki, Ph.D.
Chair, Department of Computer Science and Engineering

Committee on Final Examination

____________________________________
Michelle A. Cheatham, Ph.D.

____________________________________
Michael T. Cox, Ph.D.

____________________________________
Mateen M. Rizki, Ph.D.

____________________________________
Robert E.W. Fyffe, Ph.D.
Vice President for Research and Dean of the Graduate School
ABSTRACT


A cognitive system attempts to achieve its goals by utilizing the appropriate resources present to yield the best possible outcome within a short duration. To achieve the goals in such an efficient manner, it is important for the agent to manage its goals well. Goal management not only makes the agent efficient but also flexible, more durable to the sudden changes in environment, and self-reliant. Goal Management consists of various goal operations including goal formulation, selection, change, delegation, achievement and monitoring. Each operation is unique and has its own significance in aiding the performance of the agent. The thesis work focuses on the implementation of two particular goal operations. These are goal selection and goal change with concentration of the former.

Goal selection allows the agents to choose among its goals by using any criteria which is appropriate for the domain. Goal change allows the agent to change its current goal to another goal because of reasons like inadequate amount of resources or detection of a discrepancy. The implementation of these operations is done within a cognitive architecture called the Metacognitive Integrated Dual-Cycle Architecture in the two problem domains of construction and restaurant. In the construction domain, the goals are to construct the towers using the resources within a provided time limit, and in the restaurant domain, the goals are to satisfy the maximum number of people by serving items ordered with a limited amount of money. After the implementation of goal
selection and goal change, the work is evaluated using various methods, one of which is the comparison of the performance of MIDCA with and without those goal change operations and the other is by comparing two different goal selection methods. Several graphical depictions and mathematical formulae are presented that support the course of performance comparison.
# Table of Contents

Abstract .................................................................................................................................................. iii

Table of Contents .................................................................................................................................... v

List of Figures .......................................................................................................................................... vii

List of Tables .......................................................................................................................................... x

1. Introduction .......................................................................................................................................... 1
   1.1. Overview ......................................................................................................................................... 1
   1.2. Current Research ........................................................................................................................... 2
   1.3. Contribution ..................................................................................................................................... 3
   1.4. Outline of Thesis ............................................................................................................................. 4

2. Goal Operations .................................................................................................................................... 6
   2.1. Goal Selection .................................................................................................................................. 8
   2.2. Goal Change .................................................................................................................................... 11

3. Metacognitive Integrated Dual-Cycle Architecture ............................................................................. 15
   3.1. Goal Graph ..................................................................................................................................... 17
   3.2. Domains in MIDCA ........................................................................................................................ 20
       3.2.1. Construction domain ............................................................................................................... 20
       3.2.2. Restaurant domain ................................................................................................................... 21

4. Evaluation of goal selection and goal change ..................................................................................... 23
   4.1. Method to evaluate goal selection and goal change ......................................................................... 23
       4.1.1. Goal selection in the construction domain with time being constant ................................... 23
       4.1.2. Goal selection in the construction domain with varying time, score and deadline ............ 26
       4.1.3. Goal change in the construction domain ............................................................................... 27
       4.1.4. Goal selection in the restaurant domain with constant performance and money ............... 28
       4.1.5. Goal selection in the restaurant domain with varying performance and money .................. 29
   4.2. Results ............................................................................................................................................. 30
       4.2.1. Goal selection in the construction domain with constant deadline ................................... 30
       4.2.2. Goal selection in the construction domain with varying time, score and deadline ............ 34
4.2.3. Goal change in the construction domain ................................................................. 46
4.2.4. Goal selection in the restaurant domain with constant performance and money ................................................................. 48
4.2.5. Goal selection in the restaurant domain with varying performance and money ................................................................. 50

5. Background and Literature Review ................................................................................ 55
5.1. Cognitive systems and goal operations ........................................................................ 55
  5.1.1. Soar cognitive system .......................................................................................... 55
  5.1.2. ACT-R cognitive system .................................................................................. 56
  5.1.3. CLARION cognitive system ............................................................................ 58
  5.1.4. ICARUS cognitive system ............................................................................... 59
5.2. Other goal selection and change strategies ................................................................. 60

6. Discussion .......................................................................................................................... 62
  6.1. Future Research ....................................................................................................... 63
  6.2. The Conclusion ....................................................................................................... 64

References ............................................................................................................................. 66
List of Figures

Figure 1. Example Problem in Construction domain ................................................................. 9
Figure 2. Metacognitive Integrated Dual-Cycle Architecture ................................................. 16
Figure 3. Example of goal graph structure ............................................................................... 18
Figure 4. Results with no time limit ......................................................................................... 30
Figure 5. Arbitrary Vs. Intelligent selection when deadline=5 seconds. .................. 31
Figure 6. Arbitrary Vs. Intelligent selection when deadline = 10 seconds. .............. 32
Figure 7. Arbitrary Vs. Intelligent selection when deadline = 15 seconds. .......... 33
Figure 8. Arbitrary Vs. Intelligent selection when score and time vary by 20% and deadline = 5 seconds and remains constant. ......................................................... 34
Figure 9. Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 5 seconds and remains constant. ......................................................... 35
Figure 10. Arbitrary Vs. Intelligent selection when score and time vary by 20% and deadline = 10 seconds and remains constant ..................................................... 36
Figure 11. Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 10 seconds and remains constant ..................................................... 37
Figure 12. Arbitrary Vs. Intelligent selection when score and time vary by 20% and deadline = 15 seconds and remains constant ..................................................... 38
Figure 13. Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 15 seconds and remains constant ..................................................... 39
Figure 14. Arbitrary Vs. Intelligent selection when score and time vary by 20% and deadline = 5 seconds but varies by 20%................................................................. 40

Figure 15. Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 5 seconds but varies by 50%................................................................. 41

Figure 16. Arbitrary Vs. Intelligent selection when score and time vary by 20% and deadline = 10 seconds but varies by 20%................................................................. 42

Figure 17. Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 10 seconds but varies by 50%................................................................. 43

Figure 18. Arbitrary Vs. Intelligent selection when score and time vary by 20% and deadline = 15 seconds but varies by 20%................................................................. 44

Figure 19. Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 15 seconds but varies by 50%................................................................. 45

Figure 20. Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 15 seconds but varies by 50%................................................................. 46

Figure 21. Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 15 seconds but varies by 50%................................................................. 47

Figure 22. Arbitrary Vs. Intelligent selection when budget is $20. ......................... 48

Figure 23. Arbitrary Vs. Intelligent selection when budget is $50. ......................... 49

Figure 24. Arbitrary Vs. Intelligent selection when score and money vary by 20% and budget is $20. ................................................................. 50

Figure 25. Arbitrary Vs. Intelligent selection when score and money vary by 50% and budget is $20. ................................................................. 51
Figure 26.  Arbitrary Vs. Intelligent selection when score and money vary by 20% and budget is $50. ................................................................. 52

Figure 27.  Arbitrary Vs. Intelligent selection when score, money vary by 50% and budget is $50. ................................................................. 53

Figure 28.  ACT-R Model. ........................................................................................................... 57

Figure 29.  CLARION Model. .................................................................................................... 58

Figure 30.  ICARUS Model. ...................................................................................................... 59
List of Tables

Table 1. Algorithm for goal selection. ................................................................. 11
Table 2. Formalism for goal change. ................................................................. 13
Table 3. Algorithm for beta and choose ............................................................ 14
Table 4. Goal graph interaction with MIDCA phases ........................................ 19
Table 5. Agent estimates of time taken for each tower and the blocks above them... 24
1. Introduction

1.1. Overview

Any cognitive system is designed to achieve certain goals; the goals might be provided to the agent externally by the user or formulated by the agent itself. After acquisition of all the goals, the agent has to perform several operations on the goals like, selection of the first goal to perform, modification of the selected goal and checking if the selected goal is achieved. All these operations make the agent flexible, self-reliant and improve its performance. The thesis work focuses on two operations of these, goal selection and goal change. Goal selection is the process of selecting one or more goals from the given set of goals, goal change is the process of changing the current goal to some other goal because of several reasons.

If the answer to the question “What do I want to achieve?” defines the goal of the agent, then “What do I want to achieve first among all of the goals?” defines goal selection. Goal selection is a complicated process. To achieve it, the agent has to take into consideration several factors such as the environment, existing requirements, benefits and costs. The factors considered would be domain specific and tend to change along with the domain. The factors required to perform such operations are provided by the user currently. The agent performs the goal selection operation by considering those factors to improve its performance.
The goal change operation allows an agent to change its current goal because of several reasons. For example, the goal might be no longer valid, the resources to achieve the goal might not be sufficient or there might be an emergency situation like fire in the building. In all these situations, if the agent waits for the user to specify its goal or continues on its current goal, then the agent would not be considered smart. Apart from being considered unintelligent, there is a serious threat to the agent in the case of situations like fire which cause a significant amount of property loss. So in order to be flexible and to avoid potential threats to itself and its environment, the agent should be able to perform goal change. In the thesis work goal change is achieved by tracking the amount of resources present. Initially, the best possible goal to achieve is determined by utilizing the resources present, as the agent starts to use the resources the resources start to reduce in quantity, once the agent is out of resources, then it changes its current goal to its best possible replacement within its scope.

1.2. Current Research

The current research corresponds to a field of artificial intelligence called cognitive systems, and to be more specific, goal reasoning. Goal reasoning provides the agent an ability to formulate, manage and reason about its own goals without the help of an external source. The applications of goal reasoning are vast and include several cognitive architectures, unmanned vehicles, manufacturing industries as it tries to make the agent self-sufficient. The work done by Weber et al. (2010), applying goal driven autonomy to StarCraft has implemented the goal reasoning in game building domain where the agents would react to the unexpected game events. The main components of the model used in
the work above include discrepancy detection, explanation generation, goal formulation and goal management which are implemented in a game called StarCraft. Here the author implements reactive planning using the ABL reactive planning language. There are several other relevant researches for the thesis, which are presented in Chapter 5.

The current work is implemented in a cognitive architecture named Metacognitive Integrated Dual-Cycle Architecture (MIDCA). Its architecture and functional performance is explained in Chapter 3.

1.3. Contribution

The thesis work contributes the following contributions:

1) Implements a solution to the goal selection problem;

2) Implements a solution to the goal change problem;

3) Extends the MIDCA intend module;

4) Extends the MIDCA metacognition cycle.

Goal selection and Goal change are implemented in two domains of a cognitive architecture called MIDCA. The two domains are construction and restaurant. Goal selection and goal change are complicated tasks which an intelligent being, say a human, performs on a daily basis. Many have used various approaches to perform these operations. The thesis modifies and extends the work of goal selection using information measures (Johnson et al., 2016), a domain specific selection criteria in which the agent tries to reduce its uncertainty by trying to increase its knowledge about the environment
domain. The domain here consists of one airport and two office buildings. Our work is implemented in multiple domains using a more general selection criterion. The goal change operation from goal transformations in continuous planning (Cox and Veloso, 1998) is implemented, where the predicates and objects of the goals are changed when the appropriate pre-conditions are met.

The construction domain generates a random set of goals to construct towers of height one to seven. The agent performs the goals selection on the set of goals generated. A goal change is applied to the goal set if the need to transform the goal arises. The need to change a goal occurs when there is a insufficient amount of resources present to achieve the goal. Both the operations are implemented separately. In the second domain, the agent serves the items to attain the goal of satisfying a person, and the goal selection is performed in order to achieve maximum satisfaction with a limited amount of money.

The extension of the MIDCA modules is achieved by implementing the goal selection operation in the intend module and the goal change operation in the control module of the meta cycle.

1.4. Outline of Thesis

The topics which follow the introduction are arranged in the following format. Chapter 2 is goal operations, which explains some goal operations in brief and also the goal selection and goal change with their formalisms. Chapter 3 introduces the cognitive architecture MIDCA in which both the goal operations are implemented, and Chapter 4 evaluates the performance of MIDCA before and after the two goal operations and
depicts the results obtained in graphical format. Chapter 5 contains background information regarding the project and also a literature review. Chapter 6 closes with discussion, gives a brief note on what the thesis is about and the results obtained and provides some ideas for future research.
2. **Goal Operations**

Goal operations are the actions performed on goals by the agent. One of the factors in defining the efficiency of an agent is its capability to manage its goals. There are several actions which the agent performs in the cognitive architecture on goals (Cox, Dannenhauer, Kondrakunta, 2017) and some of which are as follows:

- **Goal Selection**: Selects a single or n number of goals from all the user given and formulated set of goals. This operation is used for prioritizing all the goals of the agent, it supports organizing the goals of agent in other words. It is one of the deciding factors to determine the efficiency of the agent.

- **Goal Change**: Transforms the current goal to another goal because of several reasons. The reasons might incorporate sudden changes in the environment or the agent might be out of resources and have to check for alternate goals which would almost satisfy the goal.

- **Goal Delegation**: Transferring or giving a goal or a set of goals to either a human or another agent. This is very useful when there is a need to share the work. If the agent is heavily loaded with goals, or if the agent does not know how to plan the goal, then it might delegate it to some agent which can perform the goal or delegate the goal to a human in order to ask for a plan to achieve it.
• Goal Formulation: Generating its own goals without the involvement of the user, the goal formulation is a high level task and a serious research problem to focus on, the cognitive system MIDCA is formulating its goals when a discrepancy is detected, but for an agent to perform complete goal formulation without the supervision of a user in every situation is one of the tough task to achieve.

• Goal Achievement: Generating a plan and acting upon the plan in order to achieve the goal. This operation can be considered as one of the goal operations where the most success has been achieved to date with the help of planning and enabling the agents to work on those plans. There are various planners which can generate plans and also many plans to achieve the same goal; the efficiency is determined in choosing the best plan for the situation.

• Goal Monitoring: Monitoring the changes in the environment and updating the plan accordingly if any unexpected event occurs by an external agent. This is the most important goal operation as it continuously informs the agent of all the changes in the environment. Without this goal operation the agent might look like a broken robot, performing the wrong action even if the states has been altered already.

All the above goal operations are implemented in the MIDCA, a cognitive architecture which has been elucidated in Chapter 3. Subsection 2.1 describes the goal selection algorithm used and then discusses its formalisms. Finally, Subsection 2.2 confers about the goal change and gives its formalisms.
2.1. **Goal Selection**

Goal selection is the process of selecting one or more goals from the candidate goal set. This goal operation is quite useful to prioritize the goals of the agent by the agent itself. This operation reduces the need of a human to monitor and give each goal after the completion of the previous one, which makes the agent smart enough to choose its own goals and self-reliant.

The idea to use the metrics from domains in order to prioritize or select goals came from Johnson et al. (2016) where the metrics from the domain are used to select the goals. The domain elucidated in the work contains one airport and two office buildings, the agent tries to locate an officer in the world and the world is unknown to the agent. To be clear the goal of the agent here is to locate an officer in the uncertain world within a limited amount of time, So the agent tries to reduce the amount of uncertainty which it has in order to find the officer within a time limit. For this purpose, the author uses information metrics such as the distance traversed and time or area, to select the goals.

\[
Information measures = I = \frac{\text{max uncertainty}}{\text{deadline}}
\]

A simple ratio of the information measures is calculated and the one with a minimum ratio is selected in order to minimize the uncertainty of the location or to maximize its knowledge about the domain within a short span of time. The work is strictly domain specific and cannot be implemented for other domains.

In the thesis work, the work above is modified such that it fits in for other domains and an attempt to generalize the work is made. Selection criteria is considered in each domain.
to perform goal selection and the metrics used will act as measures of performance of the agent in that domain. Currently the researcher provides the agent with factors to be considered by the agent in a domain, this can be further improved to detect the measures by the agent itself in the future. If we talk about this in general, then the selection criteria give or provide the user with the ratio of an estimate of performance measure over an estimate limiting value related to the specific domain, as represented in the formula below.

\[
Selection \ criteria = C = \frac{\text{performance measure}}{\text{limiting factor}}
\]  

(2)

There are two domains in which goal selection is implemented, and those two are the construction and the restaurant domains. Each domain is very different from the other, and so are the performance measures. The performance measure for the construction domain is height of the tower, and for the restaurant domain, it is the number of people satisfied.

Consider an example in the construction domain as shown in Figure 1. Here the performance measure \( P_t \) for the towers B1, B2 will be \( P_1 \) and \( P_2 \) respectively, whereas

![Figure 1: Example Problem in Construction domain.](image-url)
the limiting factor of time \( t_i \) will be \( \bar{t}_1 \) and \( \bar{t}_2 \). So, the scores will be \( C_1 = \frac{\bar{P}_1}{\bar{t}_1} \) and \( C_2 = \frac{\bar{P}_2}{\bar{t}_2} \). Let us assign values for performance measure and limiting factor. For each block in the tower assign a value of one. Let the time taken to place the base block be 1 second, for the block above it the time will increase so let that be 1.2 and for the third be 1.2. Increase the value as the height increases. So, the \( C_1 = \frac{(1+1+1)}{(1+1.2+1.2)} = \frac{3}{3.4} = 0.882 \) and \( C_2 = \frac{(1+1)}{(1+1.2)} = \frac{2}{2.2} = 0.9090 \). Finally, if we want to achieve a maximum score then the tower with minimum ratio needs to be selected.

Table 1 represents the algorithm for goal selection, the head represents the inputs to the goal operation and the output. Notations used in the algorithm are explained using simple formulas. Equation 3 below represents \( \mathcal{G} \) which represents all the goals of the agent or the problem set.

\[
\mathcal{G} = \{g_1, g_2, \ldots, g_n\}
\]

(3)

Equation 4 represents the expected time taken to construct all the towers in the problem set. Assume there are ‘m’ towers in a problem set and each tower has one to ‘n’ blocks in it. The time taken to place each block is given by \( \bar{t}^\text{D}(b) \). The equation sums up the time taken for each block of every problem set, in other words the estimated time to complete all the goals in problem set.

\[
\bar{E}(n) \leftarrow \left\{ \sum_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} \bar{t}^\text{D}(b_{ij}) \right\}
\]

(4)

Equation 5 represents calculation of the estimated time for multiple problem sets. Let \( X \) define a distinct number of problem sets and \( \bar{E}(n) \) define the estimated time for each problem set.
\[ \hat{t}(X) \leftarrow \{ \psi \}_{i=1}^{\left| X \right|} f'(n_i) \]  

(5)

Table 1 also uses some other generic notations such as \( \mathcal{P}(\hat{G}) \) representing the power set of \( \hat{G} \) and \( \emptyset \) representing null set, \( p(x) \) representing the score of each problem set and \( C(x) \) represents the ratio of performance and limiting factor. Inputs to the goal selection function are all the goals of the agent and a deadline. \( A \) represents set of all possible combination of goals that can be achieved and \( \mu \) represents best subset of the goal set \( \hat{G} \).

The algorithm explains that if \( A \) is a non-empty set then the best possible goals are selected and the ratio of performance measure over limiting factor is applied to order among the selected goals. If \( A \) is an empty set, then no goal can be selected. The agent waits for new problem set.

<table>
<thead>
<tr>
<th>Table 1. Algorithm for goal selection.</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f^{se}(\hat{G}; D); g_c = { g \in \hat{G} } )</td>
</tr>
<tr>
<td>( A \leftarrow { X } X \in \mathcal{P}(\hat{G}) \land \hat{t}(X) \leq D } )</td>
</tr>
<tr>
<td>if ( A \notin { \emptyset } ) then:</td>
</tr>
<tr>
<td>( \mu \leftarrow \text{argmax}{p(A)} )</td>
</tr>
<tr>
<td>( f^{se} \leftarrow \text{max}{C(\mu)} \text{ or } \text{min}{C(\mu)} )</td>
</tr>
<tr>
<td>else:</td>
</tr>
<tr>
<td>( f^{se} \leftarrow \text{Select no goal.} )</td>
</tr>
</tbody>
</table>

2.2. Goal Change

Goal change is the process of changing the current goal to some other goals. This operation is significant for its functioning in situations where the agent is out of resources or when an undesirable state of the environment is reached. Let us say in the construction domain if the agent has to construct stable towers, but it is out of mortar, then the
possible cases are either to wait for the mortar and do nothing or change the goal from “stable-on” to “on” and continue with the construction. Stable-on builds a sturdy tower and on builds a wobbly tower. Here the change in the goal really depends on the preference of the user. Goal change is implemented with the help of two different trees called the predicate tree and the object tree. The predicate tree represents the hierarchy of predicate and the object tree represents hierarchy of objects. These two are specified while defining the world itself. So whenever the preconditions of the operation are met the respective tree is parsed and the result is obtained.

The thesis work implements three different goal transforms: identity, generalization and specialization. The identity is always true. The generalization and specialization are both predicate transforms. Generalization would change the predicate to a more generic one like “stable-on” to “on”, while the specialization would change the predicate to a more specific one like “on” to “stable-on”. Generalization and specialization are opposites; they do not show any difference in their objects after the goal is changed. Many other goal change operations which can be performed, such as abstraction and concretion, are performed on the objects, but all those are left for future implementation. The goal change is implemented with the help of several predicate and object transforms which occur when the set of preconditions are achieved. A choose functionality is implemented in order to check the preconditions and select the goal change to perform. Let us say if more than one change has the preconditions satisfied, then it maintains the order in which the changes are to be performed on a goal and performs all of them in an iterative manner.
Table 2 depicts the formalisms (Cox, Dannenhauer, Kondrakunta, 2017) for identity, generalization and specialization. The identity is always true; the generalization has three preconditions to satisfy. The first precondition specifies that the predicate should belong in the class hierarchy tree and all the objects should belong to the object class i.e., the predicates and objects should belong to the specified domain, the second precondition specifies that there should be one other predicate which is a superclass of the predicate specified in goal and the third precondition checks whether the amount of resources present are sufficient or not. If all the above three conditions are met, then the goal generalization happens, but if even one of the preconditions is not met then the goal is not generalized. Specialization also has three preconditions, the first one specifies that the predicate should belong in the class hierarchy tree and all the objects should belong to the object class i.e., the predicates and objects should belong to the specified domain, the second one states that the goal predicate should not be the leaf node and the third checks for the amount of resources.

<table>
<thead>
<tr>
<th>$\delta^e$ (gc: G) $\triangleright$ G</th>
<th>$\text{head}(\delta^e)$ = generalization</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{parameter}(\delta^e) = gc = p(obj1, obj2)$</td>
<td>$\text{pre}_1(\delta^e) = p \in CL \land obj1 \in Objs \land obj2 \in Objs$</td>
</tr>
<tr>
<td>$\text{pre}_2(\delta^e) = \exists p, p'</td>
<td>p \in CL \land p' \in CL \land p_{\text{superclass}} = p'</td>
</tr>
<tr>
<td>$\text{pre}_3(\delta^e) = \text{limitedResourcesForGoal}(s, gc)$</td>
<td>$\text{pre}(\delta^e) = {\text{pre}_1(\delta^e), \text{pre}_2(\delta^e), \text{pre}_3(\delta^e)}$</td>
</tr>
<tr>
<td>$\text{res}(\delta^e) = p'(obj1, obj2)$</td>
<td>$\text{res}(\delta^e) = p'(obj1, obj2)$</td>
</tr>
</tbody>
</table>
The choose functionality is formalized (Cox, Dannenhauer, Kondrakunta, 2017) in Table 3. The choose checks for the preconditions and once all the preconditions are met then all goal change operations whose preconditions are met are saved to a variable called delta which performs all the operations in a sequence. Next chapter discusses about the architecture of MIDCA and the domains in which the goal operations are implemented.

<table>
<thead>
<tr>
<th>Table 3. Algorithm for beta and choose.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\delta^p(g_c; G)$: G</td>
</tr>
<tr>
<td>head($\delta^p$) = specialization</td>
</tr>
<tr>
<td>parameter($\delta^p$) = $g_c = p(\text{obj1, obj2})$</td>
</tr>
<tr>
<td>$\text{pre}_1(\delta^p)$ = $p \in CL \land \text{obj1} \in \text{Objs} \land \text{obj2} \in \text{Objs}$</td>
</tr>
<tr>
<td>$\text{pre}_2(\delta^p)$ = $\exists p', p</td>
</tr>
<tr>
<td>$\land p' = (p_{\text{name}}, p, (p', A_1, p', A_2, \ldots, A_m)) \land p' \in L_c$</td>
</tr>
<tr>
<td>$\text{pre}_3(\delta^p)$ = surplusResourcesForGoal($s, g_c$)</td>
</tr>
<tr>
<td>$\text{res}(\delta^p)$ = $p(\text{obj1, obj2})$</td>
</tr>
</tbody>
</table>

\[
\delta^p(g_c; G): G \\
\text{head}(\delta) = \text{identity} \\
\text{parameter}(\delta) = g_c \\
\text{pre}(\delta) = \{\text{true}\} \\
\text{res}(\delta) = g_c
\]

\[
\beta(s:S; g_c; G): G \\
\Delta \leftarrow \text{reverse(choose}(s, g_c, \Delta)) \\
\text{if } \delta \text{ in } \Delta \text{ then} \\
\text{if } \Delta = (\delta^p) \text{ then} \quad \text{// insertion only} \\
\quad \hat{\delta} \leftarrow \{g_1, g_2, \ldots, g_c, \ldots, g_n\} \cup \delta^p \land \delta^p \lor \delta^p \\
\text{else } (\hat{\delta}_1, \hat{\delta}_2, \ldots, \hat{\delta}_m) = \Delta \leftarrow \Delta - \delta^p \quad \text{// insertion plus others} \\
\quad \hat{\delta} \leftarrow \{g_1, g_2, \ldots, \hat{\delta}_m(\hat{\delta}_1(g_c)), \ldots, g_n\} \cup \delta^p \land \delta^p \lor \delta^p \\
\text{else } \hat{\delta} \leftarrow \{g_1, g_2, \ldots, \hat{\delta}_m(\hat{\delta}_1(g_c)), \ldots, g_n\} \quad \text{// no insertion} \\
\text{choose}(s:S, g_c; G, \Delta = \{\hat{\delta}_1, \hat{\delta}_2, \ldots\}: \text{poset}): \text{sequence} \\
\text{if } \Delta = \{\} \text{ then } \text{choose} \leftarrow \{\} \\
\text{else if } \forall x \in \text{pre}(\hat{\delta}_1) \land \text{satisfied}(x) \text{ then} \\
\quad \text{choose} \leftarrow \hat{\delta}_1 \mid \text{choose}(\Delta - \{\hat{\delta}_1\}) \\
\text{else } \text{choose}(\Delta - \{\hat{\delta}_1\})
\]
3. **Metacognitive Integrated Dual-Cycle Architecture**

The cognitive architecture used to implement the work is *Metacognitive Integrated Dual-Cycle Architecture* (MIDCA) (Cox et al., 2016) (Paisner et al., 2013). Figure 2 illustrates MIDCA, which has two cycles. One cycle is cognitive, while the other is metacognitive. Figure 2 shows the cognitive cycle (in orange) below the metacognitive one (in blue). The cognitive cycle interacts directly with the environment, whereas the metacognitive cycle monitors and performs actions on the cognitive layer. Each cycle has six phases. Each phase is unique and performs its own operations. There is no fixed order in which the phases are arranged; the user can add or remove phases according to his priority. The phases of the cognitive layer are:

- *Perceive:* This phase observes the real world, detects the changes made by the agent in the real world and keeps track of those changes.

- *Interpret:* This phase gets all the goals, validates them and detects anomalies.

- *Evaluate:* This phase keeps track of the goals and checks if they are achieved or not.

- *Intend:* This phase selects a single or “n” number of goals from the given or formulated set of goals. It is the focus of this work.
• **Plan**: This phase gets all the selected goals from the intend phase and checks to see if a plan already exists to achieve the goal, if not a new plan is generated for that particular goal.

• **Act**: This phase performs the actions to achieve the goal.
The phases of the meta layer are as follows:

- **Monitor**: This phase is similar to the perceive phase but it observes the cognitive layer.

- **Interpret**: This phase detects anomalies, gives the explanation and performs goal generation.

- **Evaluate**: This is for now a pass through phase.

- **Intend**: This phase just performs the goal selection.

- **Plan**: This phase does the planning for removing a phase from the cognitive layer, add a phase to the cognitive layer and for the goal change operation.

- **Control**: This phase performs the actions of removing a phase from the cognitive layer, add a phase to the cognitive layer and for the goal change operation.

The MIDCA uses some formalisms and notations to represent its goals, some of which are represented in Figure 2 such as $\mathcal{G}$ represents the goal set which contains all the goals of the agent and $g_c$ represents the current goal, which holds the goal which the agent is working on.

### 3.1. Goal Graph

Another important unit of the MIDCA, which manages all the goals, is a data structure called the goal graph. The goal graph gives partial ordering to all the unique goals of the agent. There might be one or more roots for the goal graph tree structure. There are three
classes for the goal graph namely, goal, goal node and goal graph. The goal class represents the goal as predicate and objects, the goal node contains the Goal class object and maintains the children and parents of each goal, and the goal graph class creates the goal graph by determining the parents and children of each node. Table 4 depicts the interaction of various phases of MIDCA with the goal graph. The goal graph tree is a directed tree structure as shown in Figure 3. A child node has a lower precedence compared to its parent and all the sibling nodes have equal priority. The overall working of the goal graph can be viewed as inputting all the formulated and user generated goals and then validating all those goals and making a goal graph structure with it, then reading them again from the goal graph, achieving and removing them from the goal graph. A detailed explanation of the working of goal graph is provided below Figure 3.

![Figure 3. Example of goal graph structure.](image)

The goal graph plays a major role in the functioning of the MIDCA. It interacts with almost all of the cognitive phases of the MIDCA and performs different actions with the different phases. The Table 4 shows the interaction of various cognitive phases of MIDCA with the goal graph.
Table 4. Goal graph interaction with MIDCA phases.

<table>
<thead>
<tr>
<th>Module</th>
<th>Interaction with Goal Graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perceive</td>
<td>No Interaction</td>
</tr>
<tr>
<td>Interpret</td>
<td>Gets the goals from the user and inserts them into goal graph.</td>
</tr>
<tr>
<td>Evaluate</td>
<td>Checks to see if the current goal/goals are achieved and if so removes the goal/goals and its corresponding plan from goal graph.</td>
</tr>
<tr>
<td>Intend</td>
<td>Checks to see if the goal graph is empty, if yes skips else check if the current goal is empty, if yes then selects the goal based on strategies like FIFO or a particular selection criterion and inserts into current goal. If no, then skips. Intend also inserts the formulated goal when an anomaly is detected and places it above the root node to give it highest priority.</td>
</tr>
<tr>
<td>Plan</td>
<td>Checks the goal graph for a matching plan, if exists, it checks validity. If no matching plans or plans are not valid, generates a new plan and inserts it into goal graph.</td>
</tr>
<tr>
<td>Act</td>
<td>Iterates over the plan for the current goal in order to achieve it.</td>
</tr>
</tbody>
</table>

An empty goal graph is initialized whenever MIDCA is instantiated. Here the perceive phase does not interact with the MIDCA at all, the interpret phase generates the goals or takes the goals provided by the user, validates them and inserts them onto the goal graph. The Evaluate phase continuously checks if the goal is achieved, and if it is, then it removes the goal from the goal graph. Then the intend phase gets the goal graph from memory and then selects the goals from the goal graph and puts them on the list of current goals, it selects the goals essentially using two methods: first in first out or the goal selection using factors considered from the domains. The plan phase checks the list of current goals and if does not find any plan associated with it then it generates a plan and puts it in the plan variable of the goal graph. The act phase gets the relevant plan from the goal graph and iterates continuously through it in order to perform actions to reach the goal. The interaction of the goal graph with meta phases is just in the control phase. The control phase uses the goal graph to perform goal change operations.
3.2. Domains in MIDCA.

The goal operations are implemented in two domains of MIDCA. Both domains are different from the other, they vary in the type of goals that are to be achieved and also in the way the scores are awarded after each goal.

3.2.1. Construction domain

The construction domain is named so because the goals generated are to build towers. This domain is an extension of the simple blocks world domain. The goals to construct the towers are generated randomly in a random number, and the height of the towers varies from 1 block to seven blocks. All the goals in a single random set will be distinct in height and objects i.e., towers of the same height will not be generated by a single random set and say if a block named ‘A’ is used in one tower then it wouldn’t be used in a different tower of the same set. The above two constraints are implemented in order to reduce ambiguity and to avoid the process of demolition to construct a new tower within the same set of goals or to avoid the process of no goal scenario.

Initially all the blocks are kept in a warehouse and the construction site is empty. The user can see the construction site, but the warehouse is invisible. Whenever a problem set is generated, the agent selects all goals to be achieved by calculating the ratio of the information measures. The objects related to the relevant goals will be fetched one at a time from the warehouse and a relevant operation is performed on them. The problem set is generated continuously, and the selection process is performed for every problem set. The towers constructed previously are erased each time a new problem set is generated.
A goal change is implemented when the goal of “stable-on” is to be achieved and the agent runs out of mortar.

The operators described in this domain include stack, stack-mortared, unstack, unstack-mortared, pickup, putdown, get_from_warehouse, and put_out_fire. Each operator is unique and performs different actions. For example, the stack places one block over another block, while unstack removes one block from another block. The stack_mortared operator does the same action as stack but with mortar. The operator “pickup block” is functional only when the block is on the ground/table. Putdown is executed when the block selected should be placed on the ground/table. And finally get_from_warehouse is used to get the objects from the warehouse to the site.

### 3.2.2. Restaurant domain

The restaurant domain obtains orders from a certain number of customers at the same time. Each customer might order from one dish to all the items on the menu. There are a total of fifteen items on the menu. Each item’s cost varies from the others. Goals in this domain are to serve the customer an item. If a customer orders three different items, then it is received as three different goals, and there is a possibility to receive a maximum of eight customer orders at a time. So, in this domain if there are two customers ‘A’ and ‘B’, each of their orders may contain one or more of the same items, but both orders should not be the same. This case is taken into consideration to reduce the ambiguity.

Initial conditions of this domain include no person in the restaurant and no dishes in the restaurant. When each problem set is generated the number of dishes are prepared based
on the investment limitation, every time only the business for a fixed amount of money can be done. So the goal selection is implemented to choose the goals which yield the best customer satisfaction within the investment amount.

The operators in this domain include take_order, prepare_order, and serve_order. Take_order receives an order from a customer and marks the state of the order as pending and the order_received. Prepare_order takes the pending orders and updates the state as order_prepared. Serve_order updates the state order_prepared, removes the state order_received and serves the order.
4. Evaluation of goal selection and goal change

Here evaluation of the goal selection and goal change operations is performed using different methods. The methods used to evaluate the goal operations are clearly explained in Section 4.1, and the results are provided in Section 4.2.

4.1. Method to evaluate goal selection and goal change

The evaluation is carried out by using various methods in different cases. These methods are explained below.

4.1.1. Goal selection in the construction domain with time being constant.

The goal selection operation in the construction domain happens using two methods. The first method is FIFO (First In First Out), i.e., in which a goal is taken in the arbitrary order it happens to be entered. The second method is using the limiting factors of each domain which decides the goal. Every time MIDCA is initialized, a random goal set to construct some $n$ number of towers is generated, and MIDCA performs goal selection on the input goals using FIFO as the selection criteria. Under FIFO for a random generated list of towers, the first goal is selected and achieved and then the next and so on. For the one with selection criteria, MIDCA chooses based on a decision ratio between performance and time estimates.
**Performance function:**

A scoring function assigns each tower a performance number when the construction of a tower is completed within time constraints. Each tower is constructed by stacking \( n \) number of blocks. For a tower of height \( h \) constructed successfully within some time limit or deadline the score achieved would be \( h \). And for constructing \( m \) number of towers whose heights are \( h_1, h_2, \ldots, h_m \), the score would be \( P = \sum_{n=1}^{m} h_n \). The towers which exceed the time limit will receive a score of 0.

**Temporal Function:**

The construction of any tower takes time, and the time taken increases as the height of the tower increases because the robot needs to be more cautious when the height increases. There will not just be an increase in overall time, but there will also be an increase in time for each stack as the height of the tower is increased, and the increase would be a nonlinear function. In this implementation the estimates of time values are provided manually and stored in MIDCA. As the height of towers varies from one to seven, Table 5 shows potential estimates for construction. The time here is in seconds as this is a virtual representation, but in reality it would differ.

<table>
<thead>
<tr>
<th>Tower Height</th>
<th>Overall Time</th>
<th>Time to place upper block</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2.2</td>
<td>1.2</td>
</tr>
<tr>
<td>3</td>
<td>3.4</td>
<td>1.2</td>
</tr>
<tr>
<td>4</td>
<td>5.4</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>8.4</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>13.4</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>22.4</td>
<td>9</td>
</tr>
</tbody>
</table>
Now, the ratio of the performance function over the estimated time is calculated for all the goals in the random problems, and the tower with the minimum ratio is chosen first. This goal is achieved, then the second smallest ratio is selected and achieved and so on.

\[ C = \frac{\hat{P}}{\hat{t}} \]  

(6)

shows where \( \hat{P} \) is the estimated performance and \( \hat{t} \) is the estimated time taken to complete the goal. The evaluation method functions by taking into consideration two cases. One is with no time limit, and the other is with a particular deadline. Let \( D \) indicate the overall deadline for a particular problem.

**Case 1: With no deadline or \( D = \infty \)**

In this case as there is no time limit, all the towers in each random problem are constructed by starting to choose each one with a minimum ratio of performance measures to the maximum.

**Case 2: With deadline of \( D = X \) seconds**

In this case as there exists a time limit of \( X \) seconds, all the towers within the problem set may or may not be constructed before the deadline. The algorithm must be able to choose the best possible subset of goals which can be achieved from the problem set within the time limit of \( X \) seconds. The combination of all the goals within \( X \) seconds are listed, the summation of scores within each combination is also listed and the one with a maximum score is selected.

Consider a simple problem set of 3 towers B1, B2, B3 with the scores \( \hat{P}_1, \hat{P}_2, \hat{P}_3 \) and times being \( \hat{t}_1, \hat{t}_2, \hat{t}_3 \) respectively, the time limit being the same \( X \) seconds. Assume that
\( t_1 \leq X; t_2 \leq X \) and \( t_3 > X \). The tower B3 is eliminated in the first place as the time taken to construct the tower exceeds the limit. Now consider the other two towers and check if \( t_1 + t_2 \leq X \) if yes, then the possible combinations of towers would be B1, B2, B1+B2 and their respective scores are either \( P_1, P_2, P_1 + P_2 \) the greatest among the three is surely \( P_1 + P_2 \). So the two towers B1 and B2 are constructed. Among those, as only one tower can be constructed, the one with the least \( \rho / \ell \) is selected. Else if \( t_1 + t_2 > X \) then the possible combinations are B1, B2 and the tower with the maximum score among the two is selected and constructed. The algorithm can be better explained by considering a random problem generated by MIDCA and finding its solution through the algorithm as explained in the next section.

4.1.2. Goal selection in the construction domain with varying time, score and deadline.

In this case the actual time to finish the construction \( t \), performance function \( P \) and the deadline \( D \) are varied. Goal selection is performed with both the methods as stated above. There are the following cases in this scenario.

**Case 1: With a constant deadline of \( D = X \) seconds and varying \( t, P \)**

The deadline in this case remains constant, but the limiting factor and performance function are varied. For each action there is a particular time, say \( t_1 \), which it consumes and gains a particular score \( P_1 \) after successful completion of the action. In this case the time is not constant, but the time is a random number within the range of \( (t_1-0.2*t_1) \) to \( t_1+0.2*t_1 \) or \( t_1 \pm 20\% (t_1) \), similarly the score achieved is also a random number within
the range \((P1-0.2*P1)\) to \((P1+0.2*P1)\) or \(P1\pm20\%(P1)\). So the performance function is calculated in two ways here, one is the expected and other is the actual. The expected selection criteria function considers the values of time and scores without the variation, while the actual performance function considers the random values generated after the variation. The values are varied with both 20\% and 50\%.

**Case 2: With a varying deadline \(D\), time \(t\) and performance \(P\)**

In this case along with the variation of time and performance the deadline \((D1)\) is varied such that it can be a random number between \(D1\) to \((D1-0.5*D1)\), here the deadline is only decreased because of the fact that the performance would surely increase for higher deadlines and our focus is on small deadlines.

4.1.3. **Goal change in the construction domain.**

The goal change in the construction domain happens when the goal is to construct a stable tower and the agents has no more resources to make the tower stable, in this case the agent tries to complete the goal by replacing the goal “stable-on” with “on”. The performance of the MIDCA with and without goal change is evaluated by assigning a score of 2 for “stable-on” stack operations and 1 for “on” stack operations. The reading is recorded by varying the number of resources for the same 30 problem sets above and the final scores are awarded for each tower by counting the number of “stable-on” and “on”.

Consider the problem set of two towers \(B1\) and \(B2\) with respective heights being 5 and 6. The number of mortars present are seven. All the goals here are to construct a stable tower using mortar blocks. The construction of tower \(B1\) is started. To construct the
stable tower B1 4 mortar blocks are needed and by the completion of B1 the agent is left
with 3 mortar blocks. Now B2 will be constructed and after stacking 4 stable blocks, the
agent will be out of resources and the goal is transformed from “stable-on” to “on” and
the remaining two blocks are placed. In this scenario the score achieved will be, score for
B1 is 4*2=8, score for B2 is 3*2+1+1= 8. Therefore, the overall score is (4+3) *2+2 =
16. For the same scenario but without goal change the agent will stop the construction
when the agent is out of resources, so the score achieved for B2 will be 0 as it is not
complete and the score achieved for B1 will be 4*2 =8= final score. The goal selection
and goal change are evaluated as separate tasks currently.

4.1.4. Goal selection in the restaurant domain with constant performance and
money.

Goal selection in the restaurant domain is also done using FIFO and the selection method
using factors from domains. Selection with FIFO is similar to the construction domain,
but with the other selection method the performance function and the limiting factor or
the temporal function changes. Here the limiting factor is money. The calculation of both
the functions is as depicted.

Performance function:
A scoring function assigns the score for each customer based on the number of dishes he
ordered, for each dish/item a customer orders the score assigned is ‘1’. So, if a customer
orders m items say $i_1, i_2, \ldots, i_m$, then he will be awarded a score or satisfaction measure of
\[ \hat{P} = \sum_{n=1}^{m} i_n = m \] if all the items he ordered are served within the money limit, else his
satisfaction score will be ‘0’. No partial scores are assigned for partially completed orders.

**Money Function:**

Each item has been assigned a cost, for example waffles cost $2, cookies $2, hashbrown $1. So, if a customer orders “n” items which cost \( m_1, m_2, \ldots, m_n \) respectively, then the overall cost \( \bar{m} = \sum_{n=1}^{n} m_n \). Even for calculation of total money the partial orderes are not considered.

Now the performance ratio is calculated using \( P \) and \( \bar{m} \), \( C = \frac{P}{\bar{m}} \). The order with the maximum ratio is selected first in order to ensure maximum customer satisfaction. In both the domains as orders are known to the agent beforehand the agents selects all the combinations within the limiting factor. Then the ratio is applied to select within the combinations.

4.1.5. **Goal selection in the restaurant domain with varying performance and money.**

A variation of \( \pm 20\% \), \( \pm 50\% \) is introduced for both FIFO and selection methods in performance and money similar to the construction domain and the results are plotted in the Section 4.2.5.

4.2. **Results**

The results obtained for goal selection and goal change in the construction domain and the restaurant domain for various scenarios are presented.
4.2.1. **Goal selection in the construction domain with constant deadline.**

In this case the deadline is constant but the experiment is performed with three different deadlines in order to observe the behavior of the MIDCA with various deadlines.

**Case 1: With no deadline or \( D = \infty \)**

![Figure 4. Results with no time limit.](image)

Figure 4 depicts the case where there is no time limit and all the goals are achieved by both the methods. Hence 100% efficiency is achieved for every problem set generated but in reality this might not be possible because of which a deadline is introduced in the following cases and the behavior with a fixed deadline is analyzed.
Case 2: With a constant deadline $D = X$ seconds.

Figure 5. Arbitrary Vs. Intelligent selection when deadline=5 seconds.

Figure 5 above depicts the results comparing FIFO and selection method with a deadline of 5 seconds. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. The X axis represents the problem sets, here each problem set is the average of three different problems. The Y axis represents the average normalized score achieved for each problem set.
Figure 6 above depicts the results comparing FIFO and selection method with a deadline of 10 seconds. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. The X-axis represents the problem sets, here each problem set is the average of three different problems. The Y-axis represents the average normalized score achieved for each problem set. The scores at problem set 7 coincide for both methods.
Figure 7 above depicts the results comparing FIFO and the selection method with a deadline of 15 seconds. As the graph clearly depicts the score achieved using the performance measures is higher than using the FIFO. The X axis represents the problem sets, here each problem set is the average of three different problems. The Y axis represents the average normalized score achieved for each problem set.

Figure 7. Arbitrary Vs. Intelligent selection when deadline = 15 seconds.
4.2.2. Goal selection in the construction domain with varying time, score and deadline.

Case 1: With no variation in deadline or $D = \text{Constant}$.

Figure 8.  Arbitrary Vs. Intelligent selection when score and time vary by 20% and deadline = 5 seconds and remains constant.

Figure 8 above depicts the results comparing FIFO and selection method at the constant deadline of 5 seconds and variation in time, score being 20%. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. Even the actual score of the selection method is higher than the expected score of FIFO. Here the expected vs actual scores of FIFO are the same every time except for problem set 3, where there is a slight variation.
Figure 9. Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 5 seconds and remains constant.

Figure 9 above depicts the results comparing FIFO and the selection method at the constant deadline of 5 seconds and variation in time, score being 50%. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. The difference between the expected vs actual scores for both the methods is higher when compared to the 20%. Here the expected vs actual scores of FIFO vary by only a small amount.
Figure 10. Arbitrary Vs. Intelligent selection when score and time vary by 20% and deadline = 10 seconds and remains constant.

Figure 10 above depicts the results comparing FIFO and the selection method at the constant deadline of 10 seconds and variation in time, score being 20%. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. Here the expected vs actual scores for the selection method are same for all the cases except for problem set 1, 3 and 4.
Figure 11 above depicts the results comparing FIFO and the selection method at the constant deadline of 10 seconds and variation in time, score being 50%. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. The difference between the expected vs actual scores for both the methods is higher when compared to the 20% change of the same case.
Figure 12. Arbitrary Vs. Intelligent selection when score and time vary by 20% and deadline = 15 seconds and remains constant.

Figure 12 above depicts the results comparing FIFO and selection method at the constant deadline of 15 seconds and variation in time, score being 20%. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. There is very little change for the expected vs actual for both the methods.
Figure 13. Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 15 seconds and remains constant.

Figure 13 above depicts the results comparing FIFO and the selection method at the constant deadline of 15 seconds and variation in time, score being 50%. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. Here the expected vs actual scores of both methods vary just by small amount.
Case 2: With variation in deadline or $D = \text{Varies}$.

Figure 14. Arbitrary Vs. Intelligent selection when score and time vary by 20% and deadline = 5 seconds but varies by 20%.

Figure 14 above depicts the results comparing FIFO and the selection method at the variable deadline of 5 seconds and variation in time, score being 20%. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. Here the expected vs actual scores of both methods vary by just a little amount; they coincide for many of the problem sets.
Figure 15. Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 5 seconds but varies by 50%.

Figure 15 above depicts the results comparing FIFO and the selection method at the variable deadline of 5 seconds and variation in time, score being 50%. The graph clearly depicts the score achieved using the performance measures is higher than using FIFO. The difference between the expected vs actual scores at 50% is higher than that of 20% of the same scenario.
Figure 16. Arbitrary Vs. Intelligent selection when score and time vary by 20% and deadline = 10 seconds but varies by 20%.

Figure 16 above depicts the results comparing FIFO and the selection method at the variable deadline of 10 seconds and variation in time, score being 20%. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO.
Figure 17.  Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 10 seconds but varies by 50%.

Figure 17 above depicts the results comparing FIFO and the selection method at the variable deadline of 10 seconds and variation in time, score being 50%. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. Here the expected vs actual scores of both methods vary by just a little amount, they coincide for almost many of the problem sets.
Figure 18. Arbitrary Vs. Intelligent selection when score and time vary by 20% and deadline = 15 seconds but varies by 20%.

Figure 18 above depicts the results comparing FIFO and the selection method at the variable deadline of 15 seconds and variation in time, score being 20%. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. Here the expected vs actual scores of both methods vary by a significant amount for both the methods. The expected overall efficiency of the selection method is greater than 50% whereas the actual efficiency of the selection method is 40%, and for FIFO the percentages are around 30% and 25%.
Figure 19. Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 15 seconds but varies by 50%.

Figure 19 above depicts the results comparing FIFO and the selection method at the variable deadline of 15 seconds and variation in time, score being 50%. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. Here the expected vs actual scores of both methods vary by just a little amount, they coincide for almost many of the problem sets.
4.2.3. Goal change in the construction domain.

Figure 20. Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 15 seconds but varies by 50%.

Figure 20 above depicts the results or the scores achieved by MIDCA with no goal change for varying number of mortars. Each problem set is an average of three problem sets. Minimum score achieved here is when the available mortar is five and the efficiency of the agent was 20% and, the maximum score achieved is at the number of mortars being 20, efficiency achieved at this point is 100% as 20 mortars are sufficient to make all the towers stable.
Figure 21. Arbitrary Vs. Intelligent selection when score and time vary by 50% and deadline = 15 seconds but varies by 50%.

Figure 21 above depicts the results achieved by MIDCA with goal change for varying number of mortars. Each problem set is an average of three problem sets. Minimum score achieved here is when the available mortar is five and the efficiency of the agent was around 70%. Here we can clearly observe that the efficiency of the agent with goal change has significantly improved with the same number of resources for the same problem sets. The maximum score achieved is at the number of mortars being twenty, efficiency achieved at this point is 100% as twenty mortars are sufficient to make all the towers stable.
4.2.4. Goal selection in the restaurant domain with constant performance and money.

Figure 22. Arbitrary Vs. Intelligent selection when budget is $20.

Figure 22 above depicts the results comparing FIFO and the selection method at a budget limit of $20. As the graph clearly depicts the score achieved using the selection criteria is higher than using FIFO. The X-axis represents the problem sets, here each problem set is the average of three different problems. The Y-axis represents the average normalized score achieved for each problem set.
Figure 23. Arbitrary Vs. Intelligent selection when budget is $50.

Figure 23 above depicts the results comparing FIFO and the selection method at a budget of $50. As the graph clearly depicts the score achieved using the selection criteria is higher than using FIFO. The X-axis represents the problem sets, here each problem set is the average of three different problems. The Y-axis represents the average normalized score achieved for each problem set.
4.2.5. Goal selection in the restaurant domain with varying performance and money.

![Graph depicting performance comparison]

Figure 24. Arbitrary Vs. Intelligent selection when score and money vary by 20% and budget is $20.

Figure 24 above depicts the results comparing FIFO and the selection method by varying performance and money. The variation here is 20%. The budget is $20. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. Here the expected vs actual scores using the selection criteria vary, whereas for FIFO they coincide for almost all of the problem sets.
Figure 25. Arbitrary Vs. Intelligent selection when score and money vary by 50% and budget is $20.

Figure 25 above depicts the results comparing FIFO and the selection method by varying performance and money. The variation here is 50%. The budget is $20. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. Here the expected vs actual scores using both the methods vary. Even with the variation, selection criteria achieved 30% and FIFO achieved around 25%.
Figure 26. Arbitrary Vs. Intelligent selection when score and money vary by 20% and budget is $50.

Figure 26 above depicts the results comparing FIFO and the selection method by varying performance and money. The variation here is 20%. The budget is $50. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. Here the expected vs actual scores using the selection criteria vary, whereas for FIFO they coincide for almost all of the problem sets. In this case the actual score using the selection criteria coincides with the score of FIFO.
Figure 27. Arbitrary Vs. Intelligent selection when score, money vary by 50% and budget is $50.

Figure 27 above depicts the results comparing FIFO and the selection method by varying performance and money. The variation here is 50%. The budget is $50. As the graph clearly depicts the score achieved using the performance measures is higher than using FIFO. Here the expected vs actual scores using both the methods vary, even with the variation, selection criteria achieved is greater than 75% and FIFO achieved around 70%.

Some generic deductions can be made from observing the graphs obtained in both the domains under various conditions. Conclusions can be drawn from the graphs by diving them briefly into two categories, one is when the deadline is comparable to the limiting factor and the other is when the deadline is not comparable to the limiting factor.

The case where the deadline is comparable to the limiting factor can be observed in the construction domain where the deadlines are 5, 10, 15 seconds and the maximum limiting factor value is 22.4 seconds for the tallest tower with seven blocks. So the graphs obtained with these deadlines resulted in random values or shaped without a specific
pattern. Even the spread between the expected values of FIFO and selection method are random.

The case where the deadline is greater to the limiting factor can be observed in the restaurant domain where the deadlines are $20, $50 and the maximum limiting factor is $9. Here the expected values of the FIFO and selection method seem to converge when the deadline varies from $20 to $50 this is because of the fact that both the methods have ample of resources to achieve almost all the goals present in the problem set.
5. **Background and Literature Review**

In general, for any cognitive architecture to be intelligent and self-reliant it should perform goal operations, like formulation, delegation, selection, transformation and achievement, rather than just getting the instructions about goals from external agents. The agent must be independent and be ready to handle unspecified situations like a building on fire. Let us now take a look at some of the cognitive architectures developed, which perform some goal related operations in one way or the other.

5.1. **Cognitive systems and goal operations.**

The thesis work is implemented in the cognitive architecture MIDCA. This is not the very first cognitive architecture to come into existence. There were many in the past which tried to imitate human cognition. Some of such architectures are Soar, Act-R, CLARION and EPIC. Each architecture is different from the others and have been around for a long time.

5.1.1. **Soar cognitive system.**

The soar architecture (Liard et al., 1986) performs the decision making process through problem space, operators and states to satisfy a goal. In soar there are goals which have sub goals associated with them. Soar has been around from a very long time and has
undergone many changes in its architecture. It has versions soar1, soar2…, till the recent soar9. There are several hypotheses for Soar about the structures which represent the intelligence. There are several components in its architecture, but the relevant one to the thesis is its application of reinforced learning, where the agent learns to allocate numeric values to its operators. A value of reward is assigned based on which the agents prioritize among its operators, and if the rewards are not sufficient to select an operator then the agent generates an impasse. In order to solve this impasse a new sub-state arises in memory with its goal being to solve the impasse, then additional searching or knowledge gaining operations are performed in order to gain a solution for the impasse. Finally, when the impasse is resolved using complex reasoning mechanisms all the rules are stored in memory in order to avoid the generation of same impasse in the future. Here the impasse can be related to an anomaly in MIDCA, and assigning the reward or score can be implemented to prioritize or select among goals.

5.1.2. ACT-R cognitive system.

Another cognitive architecture is Adaptive Control of Thought–Rational (ACT-R). It has been around since 1973 and has been written in LISP. Unlike Soar, ACT-R focuses on human cognition and human intelligence, tries to understand them closely and attempts to reproduce the observations. The scientists try to understand human behavior by various physiological tests. In general ACT-R looks like a programming language. It is a framework to create various models, which add the assumptions of the user to the agent.
As shown in Figure 28, the ACT-R model is created using deductions from the psychology experiments conducted on humans together with some information about the domain. The evaluation of ACT-R is done by comparison of the accuracy of results, time taken to perform the operation and from the neurological data of FMRI. One of the implementations of ACT-R which has been very successful is “the cognitive tutor for mathematics”. One of the work relevant to the thesis here is *an integrated model of cognitive control in task switching* (Eril M. Altmann et al., 2008) which discusses how humans control their everyday tasks and switch between those tasks. The author develops a cognition control model for these at an abstract level, and then links six basic behavioral effects to the model. They have finally compared the results of task switching with task repeating and have depicted them in graphical format. The agent has shown an increase in its performance through the task switching.
5.1.3. CLARION cognitive system.

Next is CLARION, Connectionist Learning with Adaptive Rule Induction On-line. CLARION has four subsystems which have their own roles.

- **Action-centered subsystem**: This controls all types of actions being performed.
- **Non-action-centered subsystem**: This maintains the knowledge of the agent.
- **Motivational subsystem**: This provides the inspirations for perception, action and cognition.
- **Meta-cognitive subsystem**: This monitors and modifies all the other subsystems.

![CLARION Model](https://commons.wikimedia.org/wiki/File%3AClarion_Cognitive_Architecture.jpg)
Figure 29 depicts the architecture of CLARION with its four subsystems. (Sun, 2007) In the meta-cognitive subsystem the architecture does not behave like a single minded system. It has the flexibility to choose its own behavior, by collecting all available information through other subsystems and interacting with them.

5.1.4. ICARUS cognitive system.

The ICARUS architecture took some of its assumptions from the Soar and ACT-R cognitive systems. The ICARUS architecture stands alone from other systems through its primary focus on perception and action to develop a cognitive system.

Figure 30 above depicts ICARUS, its memories and the processes. The modules of ICARUS include learning, problem solving, skill execution, and conceptual interface. All the modules are cascaded and the lower level modules help or provide their outputs to the higher level modules. ICARUS (Langley & Choi, 2006) also performs the goal selection...
operation from a given set of goals by assigning priority values to each goal. The range of the values vary from 0 to 10, the value 0 indicates least priority and the 10 indicates high priority. However, the author did not put much focus on how the values are assigned.

5.2. Other goal selection and change strategies.

GRIM (Johnson et al., 2016), Goal Reasoning with Informative Measures is a system in which some information metrics like distance traversed and time are used to perform the goal selection operation. This work is used in the thesis and it is generalized for the blocks world domain in the MIDCA architecture. GRIM also presents the life cycle of a goal from (Roberts et al., 2015). T-ARTUE (Jay Powell et al., 2011) performs interactive learning to learn the knowledge of goal selection from a user. T-ARTUE not only learns through the expert, but it also accepts criticism for a wrongly selected goal and corrects so as to not repeat it in the future.

Goal operation knowledge is not just limited to cognitive architectures, but also used in many other applications like in space (Chien et al., 2005) where the goals can be triggered based on the outcomes of the previous goals. The above work inspires the work of (Rabideau et al., 2009) developing an algorithm for goal selection with oversubscribed resources, In the algorithm the constraints and priorities define which goal to select among the set of all goals. In (Wilson et al., 2013), Goal Driven Autonomy is applied to underwater unmanned vehicles where the vehicle is left to explore the undesirable places for the humans. In such places it is very important for the vehicle to formulate, prioritize and assign the goals dynamically. Even though the author did not throw much light on
how the above operations are done specifically, he discusses how the operations play a major role for the vehicle to adapt to sudden situations which might occur. Similarly, Dora the explorer, (Hawes et al., 2010) explores all the world to fill its gaps in spatial knowledge because of curiosity. Here the priorities for the goals are set by the user manually to select some goal.

Apart from the goals we can also extend to an autonomous system which automatically detects and explains the discrepancies during execution (Klenk et al., 2015). Both Titan (Williams et al., 2003) and Kirk (kim et al., 2001) choose their actions by tracking the system state using a declarative specification of the system behavior.
6. Discussion.

This thesis work focuses on the implementation of two goal operations, goal selection and goal change in a cognitive architecture MIDCA. MIDCA is a dual cycle architecture with six different phases in each layer, where each phase performs a unique functionality. The two goal operations are implemented in the construction and restaurant domains. Goal selection is the process of selecting one or more goals from the existing set of goals, while goal change is the process of changing the current goal to some other goal because of various reasons like insufficiency of resources or changes in the environment. The performance of MIDCA with these added goal operations is evaluated and depicted in a graphical format.

The goal selection operation is performed using First In First Out (FIFO) or an alternative method, performance measures. Both of the methods are implemented and are compared. The evaluation is performed using various time limits. The performance of both the methods coincide for the case where time is infinite or when the time limit exceeds the required time to complete all the generated goals, and for the remaining time limits the method using performance measures yielded better results than FIFO. The evaluation is performed for three different deadlines: 5 seconds, 10 seconds, 15 seconds in the construction domain. The evaluation for the restaurant domain is performed when the budget limit is $20 and $50. A Graph for no time limit or when the provided time is
infinite is also depicted. The evaluation is also repeated by varying the performance factor limiting factors and deadlines at different percentages.

The goal change operation is implemented by monitoring the resources required to achieve the current goal. Once when the agent is out of resources and it has no source to attain the required resources then the goal change operation takes place. In the construction domain the resources are mortar. Towers built with mortar are strong but the towers built without mortar are unsteady, so the blocks stacked using mortar are assigned a score of 2 but the ones without mortar are assigned 1. Evaluation of the goal change operation is done by recoding the scores for towers constructed. This is repeated for various counts of mortars. The values obtained without the goal change operation are also recorded and then both are compared. Results for both the operations are plotted and are depicted in graphical format. The method using performance measures yielded better results for the goal selection, and MIDCA performed better using the goal change operation.

6.1. Future Research.

The thesis work has a lot of scope and can be further continued in many directions, some of which are as follows:

- The research work can be extended in a way such that the user does not have to provide the performance measure and limiting factor to the agent, but the agent should be able to analyze the domain and select them by itself.
• Various methods to perform goal selection can be implemented and compared to the selection criteria used.
• Complexity analysis can be performed.
• Performance of the agent when the goals change dynamically can be implemented in the future.
• Qualitative factors which aid in the goal selection can also be included in the selection criteria.
• The formula which is being used by the selection criteria can be further modified by including other limiting factors and scores.
• In the goal change not all the transforms are implemented; the other transforms can be implemented.
• Other factors which cause a goal change operation apart from insufficient amount of resources can be taken into account and can be implemented.
• The evaluation can be extended to the case where the goal change and selection occurs simultaneously.
• The two goal operations can be implemented in other domains.

6.2. The Conclusion.

I see the field of artificial intelligence making its mark around the world with innovations like personal assistants, self-driving cars, and Siri. It fascinates me to watch the growth of the field and to think about its future developments. One of the goals of artificial intelligence which remains unachieved is to develop an intelligent brain. This goal of artificial intelligence has mainly been developed under a branch called cognitive systems.
A cognitive system is a system to perceive and perform in an intelligent manner on the environment to achieve its goals. Any cognitive system in one or another way attempts to imitate a human, and we as humans can see that we constantly try to reach some goals. The goals may not always be complex ones like to create an intelligent brain or to become a president, but also simpler ones like to reach a particular location in the same room, get a bottle of water. A human generates most of his/her goals. When an agent is able to perform the thought process similar to that of human in a constructive way, then this could be of a great help to the mankind.
References.


