

ATTENTION SWITCHING DURING INTERRUPTIONS

YI ZHANG, HÉLÈNE PIGOT, ANDRÉ MAYERS

Département d'informatique, Faculté des Sciences, Université de Sherbrooke, Sherbrooke, QC, J1K 2R1 Canada
E-MAIL: yi.zhang@dmi.usherb.ca, helene.pigot@usherbrooke.ca, andre.mayers@usherbrooke.ca

Abstract:

The primary role of attention is in the control of action. A model of attention is presented. In this paper, the model takes the entire Norman & Shallice's theory into account and mainly concentrates on the interactions between Contention Scheduling and Supervisory Attentional System. An AI agent is imported as an implemental approach for the model. Simulations are chosen based on different cases to demonstrate how the model reacts to random interruptions while executing a routine action.

Keywords:

Action; Attention; Interruption; Norman & Shallice theory; Contention Scheduling; Supervisory Attentional System; Schema; Activation value

1. Introduction

In daily life, people carry out actions to fulfill expected goals. An action is often suspended and resumed later because of interruptions. Depending on the main action's hazard and the interruption urgency, the two involved actions will be processed in different order. An intelligent system that takes such actions needs to represent the action in process, the changes in the environment leading to an interruption and the decisions controlling the actions order.

In psychological literature, attention is responsible for the switch from a well-learned action to a willed one. Interruptions, therefore, are under control of attention. Norman and Shallice [1] have addressed two mechanisms responsible for the two levels of control: Contention Scheduling (CS) responsible for well-learned actions, and Supervisory Attentional System (SAS) required when the sequence of actions is new, when the action itself is dangerous, or when the plan must be modified according to an unanticipated environmental change.

Cooper and Shallice [2] have implemented a detailed computational model of CS. Shallice and Burgess [3] proposed an outline of the processes and their interactions involved in SAS. As an intelligent agent, SAS is able to plan, generate strategies and solve problems. Thus,

Glasspool [4] proposed to use Domino [5], an AI agent theory, to implement the SAS. But still no implementation resumes the SAS role neither the interactions between the SAS and CS, which are essential to explain how to deal with interruptions.

We present here the implementation of the whole theory of SAS and CS and the interactions between these two mechanisms regarding interruption control. An example is displayed to highlight the processes involved. Results of the model simulations are discussed in accordance with the real world.

2. Preliminary

2.1. Definitions

Action

Action is defined as the act of purpose to accomplish a goal in some stages, usually over a period of time. Whether the purpose is conscious or not, the action is directed by a goal that is satisfied with the action completion. Some complex goals are divided to more simple ones, each of them leading an action to be performed [2]. Figure 1 shows two actions and their intended goals.

Goals management is necessary in the concurrence of actions because two actions could not always be performed during the same time. The interruptions, which are presented later, are an example of such conflict.

Attention

Attention is the ability to focus on an action. Some actions require attention: the novel actions, the dangerous ones or the actions that need deliberate plans. But the routine actions, called automatic actions, which have been performed several times in the same conditions, require much less attention. When unanticipated interruptions or unexpected environment changes occur, the attention raises to cope with the new situation. Therefore, the same action may belong to the automatic or to the willed action depending on the environment in which it is performed. For example, driving a car is considered as an automatic action.

Everyday, one drives from home to office and the way back. Through this well-known route, the driver could talk or listen to the radio while driving without much attention. But if a hazardous situation takes place in the route, like a traffic jam or a child crossing the road, the driver stops his conversation and turns his attention to the road. In addition, if the journey is new, the driver needs to pay attention on the road situation, the traffic lights, the road signs and so forth. In these cases, the action of driving is considered as a willed action.

Interruption

An interruption is defined as an unanticipated issue rising up from the environment while a main action is being performed.

Facing an interruption issue, individuals mainly react in three different manners depending on the urgency. First, if the interruption is more urgent than the current action, one switches his attention from the current action to perform the interruption action and completes the main action afterward. Second, the interruption is urgent, but the main action is currently dangerous, one still executes a part of the current action in order to keep it safe and then accomplishes the interruption. Third, the interruption is less urgent than the main action and will be processed later after the completion of the current action.

2.2. Objective of this paper

To deal with interruptions during action completion implies to deal with attention. Norman and Shallice [1] provide a psychological model composed of two mechanisms of action control, the CS and SAS. Their model is like a multiple intelligent agent. CS is implemented, but lack of precision upon the SAS prevents its implementation. Our objectives are then threefold: implementing the SAS using an intelligent agent; implementing the attention switch from the SAS to the CS; and implementing the interruptions management.

3. Theoretical and Empirical background

3.1. Norman & Shallice model

Norman and Shallice provided a framework comprised two distinct mechanisms to control actions [1]. Two types of action are considered, automatic actions and willed actions. The former refers the activities that can be carried out with diminished awareness and is under control of CS. The later regards the performances with will or attention and is under the control of SAS.

Contention Scheduling (CS)

The CS controls the well-learned or routine actions. Actions are represented as a set of schemas in CS. A schema controls an action, either an internal processing or an external movement of effectors. The core of CS is the schema hierarchy and within which those schemas are competing. An activation value is assigned to each schema. Whenever a schema's activation value is higher than a trigger threshold and higher than any of its competitor, it is selected.

Supervisory Attentional System (SAS)

The SAS controls actions under non-routine situations. Norman and Shallice suggested that SAS oversees the performance of action and participates in control by creating schema or by modifying activation values when needed [1]. Attention is required under non-routine situations. It is put onto CS through SAS, which controls over CS schemas by modifying the activation values, either increasing the values of desired schemas or inhibiting the values of improper schemas. Shallice and Burgess addressed three stages of SAS to respond to an uncommon situation [3]: the construction of a temporary new schema; the implementation of this temporary new schema; and the monitoring of the schema execution. The temporary new schema constructed and implemented by SAS replaces the source schema to control the schemas in CS to provide a procedure of dealing with the novel action.

3.2. Contention Scheduling Implementation

The CS is comprised of a schema and a selection mechanism, which is dedicated to the selection between these schemas.

Cooper and Shallice have implemented a detailed computational model of CS [2]. In the Schema Network, nodes are organized hierarchically according to different level of actions. Since an action is an act of will, each schema is also associated with a goal. The Selection Process connects the Schema Network and Motor System and oversees their status.

Each schema has an activation value, which is influenced by five factors: external influence, internal influence, self influence, lateral influence and random noise [2]. A schema is selected when its activation exceeds a threshold. This selected schema in turn excites its component schemas. Selection of a lowest-level schema will lead the execution of an action in the world.

In addition, the Schema Network cooperates with two other networks: an Object Network and a Resource Network, which respectively represent objects (e.g. a coffee mug) and resources (e.g. a hand). Nodes in Object Network and Resource Network are also associated with activation

values.

3.3. Supervisory Attentional System Implementation

According to Norman and Shallice theory, in case of a novel or dangerous situation, or when some unexpected interruption shows up, attention is required. Attention is put onto CS through SAS by modifying the activation values according to the situation.

Based on Shallice and Burgess's description [3], Glasspool has proposed Domino [5] as the approach to implement the SAS [4]. Domino is an artificial intelligent agent, which is able to react and reason about issues coming out in its environment, to raise goals, to construct plans and execute actions to achieve the goals under the current environment [5]. These processes specified in Domino are similar to those included in SAS, as listed in Table 1.

Actually, Glasspool has mapped the outline of SAS directly onto the Domino outline. Based on this mapping, Glasspool have implemented a model of SAS in Cogent [4,6]. Cogent is a visual design environment for cognitive modeling [7]. In Cogent, the models use buffers to store information and a set of rule-based processes to operate upon information.

Table 1. The main processes and their functions in SAS

SAS Component	Process Function
<i>Perception</i>	Senses any environmental change and transfer useful information to the subject.
<i>Monitoring & Goal Generation</i>	Monitors action execution and sets goals if necessary.
<i>Strategy Generation</i>	Generates different strategies to achieve the set goal
<i>Evaluation</i>	Evaluates the strategies and selects the most suitable one currently.
<i>Schemas Construction</i>	Constructs a temporary new schema for the select strategy.
<i>Beliefs Update</i>	Updates beliefs of the environment.

4. The Interruption Model Implementation

4.1. Simulations

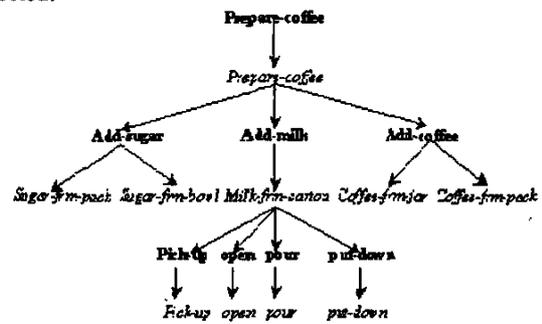
In our implementation, we continue to use the structure fleshed out by Cooper [6]. Modifications are made as few as necessary for the current purpose.

Action representation

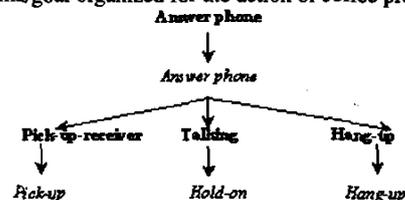
In current implementation, when encountering an interruption, the model immediate switch from the main

action to it. Two actions are taken in: coffee preparation (the main action) and answer phone call (the interruption). A parameter is introduced to identify how urgent an action should be executed. An action is represented as a pair of its name with its urgent priority like **Action (Name, Priority)** in the experimental world and the SAS. The urgent priority is an integer within interval of 1 to 100. At any time, if an action's urgent priority is higher than any others, it should be executed first.

However in CS, an action is represented as a schema, which is associated with an activation value and an intended goal. Figure 1 presents the schema/goal organization for coffee preparation and answer phone, respectively. The schemas are organized up to three levels. The leaves of the schema/goal representation correspond to the atomic actions. Each schema is associated with a goal, which consists of a set of partially ordered subgoals. For example, shown in Figure 1, the schema *prepare-coffee* is associated with the goal **prepare-coffee**, which in turn comprises of three subgoals as **add-sugar**, **add-milk** and **add-coffee**. These three subgoals are partially ordered with a constraint that the milk should be added after the other two. Only when all its subgoals are done, a schema's intended goal will be achieved. In reverse, each goal is related with several schemas, each of which identifies an approach to achieve the goal. For example, the goal of **add-sugar** are related to two schemas, *sugar-frm-pack* and *sugar-frm-bowl*, each of which provides a way to achieve the current goal. So just one of these two schemas will be selected.



(a) Schema/goal organized for the action of coffee preparation.



(b) Schema/goal organized for the action of answer phone.

Figure 1. The trees relate to schemas and goals. Goals are printed in bold type and schema in italic type.

Contention Scheduling

Center of CS is the Schema Hierarchy. Two subnets are defined as *coffee* and *phone* to represent the hierarchy of coffee preparation action and answer phone action. Therefore, a schema is specified as a binary structure like schema/subnet, the first is the name of the node and the second identifies the subnet it belongs to. For example, schema *pick-up/coffee* are different from schema *pick-up/phone*, the former represents the action of picking up an object during coffee preparation while the later represents the action of picking up the receiver to answer a phone call.

Each schema is associated with an activation value, which varies over time within a range of [0.0, 1.0]. At the initialization, all schemas' activations are set uniformly within the interval [0.05, 0.15]. As it is discussed in [2], competitive schemas inhibit each other. For the purpose that the lower-level schemas could get enough excitation to be selected out, this lateral influence parameter is set to 0.15. The self influence is set to 0.50. Schema activation is also excited by the internal influence. The internal influence, defined as the top-down influence, is calculated as follows. Let I be the internal influence on a node, let A be the activation value of its parent and let N be the number of the children of that parent,

```

if (the schema is directly triggered by SAS)
    then  $I = 1$ 
    else  $I = A/N$ 
    
```

The environment is implemented without resource network and object network. We remove the object network and resource network from CS by assuming that at any time when the subject needs allocate some objects or resources to take out an action, they are available and can be used properly.

A process named *Select & Act* in CS is in charge of selection of the most proper schema and carries out an action when a lowest-level schema is select. When a schema's activation value exceeds a threshold, which is set to 0.60, and is higher than any of its competitor's activation value, it is selected. Competitor schemas are those share the same goal or share one or more subgoals in common. For example, in Figure 1, *schema sugar-frm-bowl* is a competitor of the *schema sugar-frm-pack*, because they share the same goal **Add-sugar**. After achieving its goal, a schema should be inhibited (sending a negative exciting value) and be deselected.

Role of SAS in the interruption

Among the processes listed in Table 1, we highlight

two of them, *Perception* and *Monitoring & Goal Generation*, which play the most important role during the response to interruptions.

In this implementation, it is assumed the subject wants to prepare a coffee at the initial time. The first process in SAS, *Perception*, adds a request of the coffee preparation action in the working memory. Then *Monitoring & Goal Generation* set up a goal for it. The *Strategy Generation* in turn generates a possible approach to achieve the goal. After evaluated, this strategy then is selected to construct a temporary new schema for coffee preparation. The temporary new schema triggers the schema of coffee preparation action in the CS. So far, SAS delegates control of action to CS.

Now the coffee preparation action is controlled by the CS unless an interruption occurs. At this time *Perception* acts as an "advisor" to tell the change in the environment. When *Perception* perceives the phone rings, it feeds the request to answer phone. Then a central function in *Monitoring & Goal Generation* is responding to the determination of attention switching during interruptions. Let **Action (C, Uc)** and **Action (I, Ui)** represent the current action and the action for the interruption respectively, the function is described as following:

```

{If  $U_i > U_c$ 
Then
    set a goal for I in Goals;
    marker (stop(C)) in working memory;
    {If finished (I)
    Then resume (C)}
Else
    {If finished(C)
    Then set a goal for I in Goals}
}
    
```

The function first compares the new interruption action's urgent priority with that of the current action. If it is higher, another rule in the process will be fired to raise a goal for the new action and pause the current action at the same time. This new goal will leads another flow of processes in SAS and reconfigures CS to adopt another schema hierarchy for the action of answer phone. After finishing the phone call, the cue of an unfinished action of coffee preparation reminds the subject to resume it. Here, the unsatisfied goal is retrieved from the working memory. A goal then should be set again to continue the former coffee preparation action. The processes flow of SAS will finally reconfigure CS to handle it till its completion unless another interruption occurs. Cues in the world help the subject to determine where to continue the former action.

4.2. Results

In this section, results of the simulation are reported based on two experiments, which will be done in Cogent. The time when the phone call interruption occurs is set randomly. Cogent provides the Activation Graph of the *Schema Hierarchy* to give a dynamic view of nodes' activation value, which updates over time. Figure 2 provides a view of the schemas' activations at the time of answering the phone. While answering the phone, the initial action of coffee preparation is paused. Therefore, the schemas belong to the subnet of coffee are inhibited SAS.

Results are presented in groups of cycles to show how attention controls actions. Some representational cycles are listed out and explained. Finally, results are explained by their accordance with the real world.

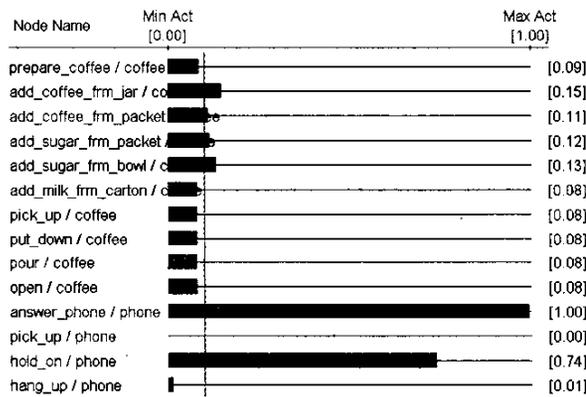


Figure 2. The histogram in Activation Graph shows schemas' activation changing over time.

Experiment 1

The answer phone action is fed in at cycle 33.

From cycle 1 to cycle 16, preparing and starting executing the action of coffee preparation.

From cycle 16 on, the current action is coffee preparation.

On cycle 41, *schema add-coffee-frm-packet/coffee* is selected.

On cycle 49, *schema add-sugar-frm-packet/coffee* is selected.

From cycle 33 to cycle 38, preparing for answer the phone.

From cycle 50 to cycle 121, executing the action of answer phone.

From cycle 124 to cycle 139, resuming the former action of coffee preparation.

From cycle 139 to the stop of the model, executing and finishing the action of coffee preparation.

Experiment 2

The answer phone action is fed in at cycle 24.

From cycle 1 to cycle 22, preparing and starting executing the action of coffee preparation.

From cycle 22 on, the current action is coffee preparation.

From cycle 24 to cycle 36, preparing for answer the phone.

From cycle 45 to cycle 118, executing the action of answer phone.

From cycle 124 to cycle 135, resuming the former action of coffee preparation.

From cycle 135 on, executing the action of coffee preparation.

On cycle 157, *schema add-sugar-frm-bowl/coffee* is selected.

On cycle 163, *schema add-coffee-frm-packet/coffee* is selected.

On cycle 829, the action of coffee preparation is finished and the model stops.

From the two experiments, we see that the schema selected to achieve a certain goal is not always the same. In experiment 1, *schema add-sugar-frm-packet* is selected to achieve the goal of **add sugar**. While in experiment 2, the schema selected for this goal is *schema add-sugar-frm-bowl*.

The order of schema's selection is different in the two experiments. In experiment 1, *schema add-coffee-frm-packet/coffee* is selected before *schema add-sugar-frm-packet/coffee*. While in experiment 2, *schema add-sugar-frm-bowl/coffee* is selected before *schema add-coffee-frm-packet/coffee*. Without order constraint, it allows the subject to choose one or another because the subgoals of **prepare coffee** are partially ordered.

Notice that in experiment 1, the phone call is fed in on cycle 33 and the action of answer phone begins on cycle 50. In this duration of its excitement, the activations of some schemas in coffee preparation are still increasing. This could be explained in real world, when the phone rings while we are executing another routine action, most of time, we need a time to respond to the ringing phone and until we pick up the receiver, we finally stop our performance of the former action. Though it may be inhibited intentionally at the very second that the phone rings up.

In the implementation, SAS interacts with CS on several stages. First of all, SAS initializes a routine action (e.g., coffee preparation) and delegates it to CS to complete by executing a set of actions. Second, when an interruption (e.g., phone is ringing) comes out during the routine action performance, SAS switches attention from the former to the coming interruption by constructing and implementing a

new temporary schema for the interruption action as well pausing the former. Third, when the interruption is done, SAS updates beliefs about its situation and reconfigures CS to continue the former routine action to complete if there no other interruptions occur.

5. Ongoing work

For now, the urgent priority is assigned to a constant. But in more mature analysis, this parameter should vary according to time and to hazardous status of the action.

In our simulations, the model performs actions separately. For example, when a phone call comes during the coffee preparation procedures, it pauses coffee action to answer the phone call and goes back on coffee action after conversation. However, under many situations, people can do these two actions together in the same time. Processing resources should be taken into account to simulate synchronous performance of two actions.

On the other hand, in real world, a complex action may be disturbed by a series of interruption. For example, while a clerk is preparing his weekly work plan, the phone rings; and while he is answering the phone, he is asked to attend an emergent meeting by the manager. This aspect should be taken into account.

6. Conclusions

Dealing with context changes is often referred to reactive intelligence. More precisely, reactive planning extends the traditional planning, by adding the current context influences in the action selection [8]. In psychology, the attention is responsible of the context awareness. One of the SAS roles is to detect the environmental changes. The automatic process is then isolated from the decisions requiring much more elaborations. We have presented in this paper the first two SAS processes involved in case of interruption: the interruption identification and the action selection hold respectively by the *Perception and Monitoring & Goal Generation* process. It could be compared with the Basic Reactive Plan, which is explained by Bryson and Stein [8]. Both of the models are based on priority levels and hierarchical plans. But, the choice of the next action is embodied in the action representation in the Basic Reactive Plan. Instead, in current model, the next action is chosen by other SAS processes. The three other SAS processes explicate the reasons leading to the next

action.

The model presented in this paper is not intended to present the whole mechanism of the Norman & Shallice model. It focuses on how attention switches and handles actions when confronting an interruption. First, it has been shown how an environmental change leads to an interruption in the CS process and second, how the SAS gives back the control to the CS when the action chosen becomes automatic. The reactive intelligence could take advantages of the psychological theories that explain the adaptive human behaviour.

References

- [1] D. A. Norman and T. Shallice, "Attention to action: willed and automatic control of action," Plenum Press, New York, 1980.
- [2] R. Cooper and T. Shallice, "Contention scheduling and the control of routine activities," *Cognitive Neuropsychology*, Vol. 17, No. 4, pp. 297-338, 2000.
- [3] T. Shallice and P. Burgess, "The domain of supervisory processes and temporal organization of behaviour," *Philosophical Transactions of the Royal Society of London B*, Vol. 351, pp.1405-1412, 1996.
- [4] D. W. Glasspool, "The integration and control of behaviour: insights from neuroscience and AI," *How to Design a Functional Mind*, AISB Convention April 2000.
- [5] S. K. Das, J. Fox, D. Elsdon and P. Hammond, "A flexible architecture for autonomous agents," *Journal of Experimental and Theoretical Artificial Intelligence*, Vol. 9, No. 4, pp. 407-440, 1997.
- [6] R. P. Cooper, "Modelling high-level cognitive processes," Lawrence Erlbaum Associates, New Jersey, 2002.
- [7] R. Cooper and J. Fox, "COGENT: A visual design environment for cognitive modeling," *Action Research Methods Instruments and Computers*, Vol. 30, No. 4, pp. 553-564, 1998.
- [8] J. Bryson and L. A. Stein, "Modularity and Design in Reactive Intelligence", *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI*. Vol 2, pp.1115-1120, Seattle, Washington, USA, 2001.