$RESTAURANT REVISITED OR 'LUNCH WITH BORIS'

Michael G. Dyer

Computer Science Department
Yale University
New Haven, CT 06520

1. Introduction

Several natural language understanding systems, such as SAM [1] and FRUMP [2], use the notion of a script [7], a knowledge structure containing a stereotypic sequence of actions. Scripts are intended to capture situations in which the behavior is so stylized that the need for complex plan or goal analysis rarely arises. A prototypical example of a script is $RESTAURANT = ENTER + BE-SEATED + WAITRESS-COMES + ORDER-FOOD + FOOD-BROUGHT + EAT-FOOD + RECEIVE-CHECK + LEAVE-TIP + PAY-BILL + LEAVE.

Scripts are useful in supplying expectations during processing. These expectations represent an active context and help in such tasks as:

(1) Pronoun Resolution -- In a $RESTAURANT context, clearly the "he" in: "He left him a big tip." is the customer while "him" must be a waiter.

(2) Word Sense Disambiguation -- The expressions "ordered" and "to go" have very different meanings in a restaurant ("John ordered a pizza to go.") than in the military ("The general ordered a private to go.").

(3) Supplying Inferences -- Once a script is chosen, processing can proceed very efficiently, since missing information is automatically provided by the script. For example:

> John ordered a lobster.
> John paid and left.
>
> Q: What did John eat?
> A: Lobster.

Scripts also contain one or more paths [1], each supplying a possible alternative sequence of actions to ghost [6] (or default) path. An alternative path in $RESTAURANT includes leaving without paying because the food was improperly cooked. This path information is used to answer questions, such as:

> Q: Why didn't John eat the hamburger?

In this case, the search heuristic starts with the 'ghost path' in which the food would have been eaten. The retrieval heuristic backs up along this path until a branch point is found. Here resides the reason:

> A: The hamburger was burnt.

that an alternate path was taken.

2. Problems with $RESTAURANT

However, representing what people know about restaurants only in terms of a $RESTAURANT script is inadequate, for three reasons:

(1) Scripts were conceived as self-contained 'chunks' of knowledge. As a result, it is difficult to share knowledge across scripts. For example, a restaurant serves meals, but people also eat meals in non-restaurant situations (home, picnics). This meal knowledge should be shared with restaurant knowledge, even though the meal-server in a restaurant differs from the meal-server at home.

(2) Because scripts are self-contained, experiences occurring within one script cannot be generalized to other relevant situations. For example, if we refuse to pay for burnt food in a restaurant, somehow the knowledge derived from this experience should be available later in an $AUTO-REPAIR context, when the mechanic fails to fix the engine properly.

(3) Scripts lack intentionality. From a scriptal point of view, each event occurs next simply because it is the next event in the script. Although script-based programs know that characters initiate $RESTAURANT to satisfy hunger, they do not know why any specific event within $RESTAURANT occurs. This is analogous to answering:

> Q: Why does the diner tip the waitress?
> A: I don't know. That's just what he does in a restaurant after he's eaten.

Lack of intentionality has both advantages and disadvantages. It is certainly more efficient, since goals and plans do not have to be processed in order to predict what a character will do next. However, it is difficult to handle novel situations, where a character's reaction might be explainable if the underlying goals and motivations (for the expected event) were known [9].

For instance, a friend admitted to me that, as a child, she enjoyed pocketing the coins people left on restaurant tables. One day she was caught by her parents, who informed her that these coins were to reward the waitress for her service. Until this moment, in which my friend grasped the intentional

---

[1]This work supported in part by Advanced Research Projects Agency contract N0014-75-C-111 and National Science Foundation contract IST7918463.

significance of tipping, it had just been a "scriptal action" for her.

3. MOPa

In BORIS, an in-depth understander of narrativea [3] [5], scripts have been augmented by a claas of knowledge structures called MOPs (Memory Organization Packets) [8] .

Like scripts, MOPs encode expectations, but unlike scripts, MOPs are not isolated chunks of knowledge. Instead, each MOP in BORIS has strands which indicate how the MOP has been constructed from other knowledge sources. Each strand connects an event in one MOP to some event in another MOP. In this wsy, MOPs are overlaid with each other. Consider diagram-1:

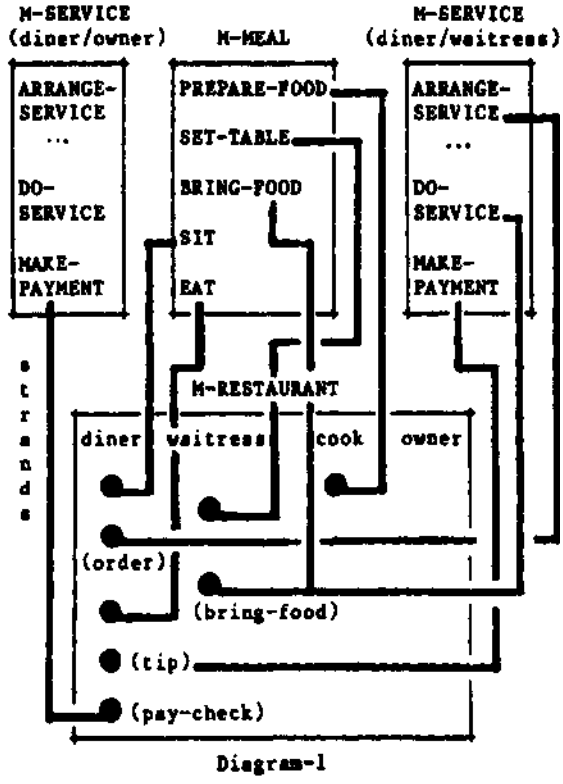

Diagram-1

Events in M-RESTAURANT ire overlaid with their corresponding events in M-KEAL and M-SERVICE. In this way, H-RESTAURANT can be viewed from different perspectives* From the perspective of M-KEAL, a restaurant is simply a setting in which people have meals. Prom the perspective of M-SERVICE, the diner in M-RESTAURANT is engaged in a aervice contract with the restaurant owner. The restaurant nuat serve food to the diner and, in return, the diner ia expected to pay for thia aervice.

As atated earlier, script-based systems represent scriptal deviation in terms of alternate 'paths'. Thus, the restaurant acript contains a BURNT-FOOD —> LEAVE-WITHOUT-PAYING path. But this approach cauaea a proliferation of paths, for two reasons: First, every poasible deviation haa to be anticipated and 'canned' into the acript; otherwise the script would not be able to handle the deviation. Second, each service-related script ends up copying the same path. For example, in $AUTO-REPAIR there has to be a BAD-REPAIR———> LEAVE-WITHOUT-PAYING path. By overlaying these scripts with M-SERVICE, a single deviation path: POOR-SERVICE———> REFUSE-TO-PAY in M-SERVICE can represent deviation knowledge at a more general level. Likewise, 'payment for service' occurs in many situations, not just in $RESTAURANT. Therefore, tipping should be understood at a morr general level than $RESTAURANT.

4. Handling Deviations

When stereotypic situations are first encountered, people let their scriptal knowledge handle the situation until a violation occurs. At this point they become aware of other perspectives associated with the script, each potentially useful in understanding the violation.

For instance, we rarely think about the contractual aspects of restaurants when eating in one. Restaurants are usually "taken for granted" as a place where one can have a meal and socialize. Only when something goes wrong do we think of the contract we're implicitly engaged in. If the service is bad, then we reconsider the amount of the tip and what it's for.

More importantly, a deviation which has never before been encountered may be handled, as long as some etrsnd exists from the script to a knowledge structure with information about this type of deviation. For inatance, if $MOVIE has strands to M-SERVICE then the very first time the movie projector breaks we can uae our deviation path in M-SERVICE to demand a refund, even if the projector breaking is a novel experience.

Consider how BORIS processes the following fragment of DIVORCE-2, a complicated narrative [4] concerning marital infidelity:

> George was having lunch ... when the waitress accidentally knocked a glass of coke on him. George was very annoyed and left refuaing to pay the check...

Briefly, an analyais of "having lunch" activates M-MEAL. When "waitress" occurs, the MOPs aaaociated with thia role are examined. If a MOP being examined has a strand to an active MOP, then it is also activated. Since there are strands from M-RESTAURANT to M-MEAL, and since M-MEAL is already active, M-RESTAURANT ia activated also.

An interpretation of "accidentally" indicates that a violation may follow. This heuristic is based upon the assumption that unintended actions usually violate scriptal expectations.

"Knocked a glass of coke on him" is analysed in terms of the Conceptual Dependency [7] primitive PROPEL (Object - Liquid). Given this event, BORIS tries to match it against the events expected in M-RESTAURANT. This match would normally fail, since M-RESTAURANT does not expect waitresses to PROPEL food-stuffs. However, "accidentally" has warned BORIS of a possible violation, so a violation match ia attempted and succeeds. At this point DORIS realisea that the PROPEL event is a violation of the event BRING-FOOD in M-RESTAURANT,

rather than some event totally unrelated to M-RESTAURAHT.

Nov what it BORIS to dot In previous systems, there would be a path in the script for such a deviation* Hovever, tbia is not the case here. When BORIS encounters a deviation, it searches the strands connected to the event where the deviation occurred. This leads to DO-SERVICE in M-SERVICE (see diagram-1).

Associated with M-SERVICE is general knowledge about how things may "go wrong" for each event in M-SERVICE. There ere several events, such as: ARRANGE-SERVICE, DO-SERVICE, INFORM-BILL, MAKE-PAYMENT, etc. For example, the sentence:

The wsitress overcharged George.

constitutes a violation of INFORM-BILL.

In addition, there is knowledge about how violations may be related to each other. This knowledge is represented by rules, such as:

If SERVER has done SERVICE
badly (or not at all),
Then SERVER should either not BILL
CONTRACTOR or BILL for amount < NORM.

If SERVER has done SERVICE badly
or BILLS CONTRACTOR for amount > NORM,
Then CONTRACTOR may REFUSE PAYMENT.

BORIS uses this knowledge to recognise the connection between the waitress PROPEL LIQUID and George's refusal to pay a check.

Left to consider is how BORIS realises that the violation of BRING-FOOD actually constitutes POOR-SERVICE. This is accomplished by tracking the goals of the characters. The PROPEL LIQUID on George is understood to cause a PRESERVE-COMFORT goal for George. This goal is examined by M-SEEVICE, which applies the following heuristic:

If SERVER causes s PRESERVATION
GOAL for CONTRACTOR while
performing SERVICE,
Then it is probably POOR-SERVICE.

Thus, BORIS uses several sources of knowledge to understand what has happened. M-RESTAURANT supplies expectations for what the waitress should have done. Knowledge about PROPEL and LIQUIDS supplies goal information, while M-SERVICE (between waitress and diner) provides very general knowledge about how contractors will respond to poor service.

5. ConeInsions

Early script-based systems operated within restricted knowledge source domains. For instance, although FRUMP had many available scripts to choose from, once a single script had been selected, the rest of the story was processed within the context of that script alone. As s result, many problems involving scriptal interactions never arose.

In this paper I have discussed various problems with scripts, and auggeated memory 'overlays' as an initial solution. This overlay scheme has several advantages:

Each knowledge atructure need know only what is directly relevant to it. For example, what a waitress does in captured in M-RESTAURANT, while her reasons for doing her job are represented at the N-SERVICE level which will handle any type of service. M-SERVICE need not be repeated for janitors, salesgirls, etc. This supports economy of storage, but more importantly, it means that any augmentation of the knowledge in M-SERVICE will automatically improve the processing ability of any MOP with strands to it.

Related knowledge eources need not be activated unless something goes wrong during processing. For instance, people do not normally think of the contract between themselves snd the restaurant manager unless they are having trouble with the service.

Finally, a given event can be understood from several perspectives. For example, a "business lunch" involves M-MEAL, M-SERVICE, M-RESTAURANT, snd H-BUSINESS-DEAL simultaneously.

ACKNOWLEDGEMENTS

Specisl thanks go to Tom Wolf, Pete Johnson, Mark Burstein and Marty Korsin for their conceptual and practical contributions to the BORIS project.

REFERENCES

[1] Cullingford, R. E. "Script Applicstion: Computer Understanding of Newspaper Stories." Tech. Rep. 116, Yale University. Computer Science Dept. 1978.

(2) DeJong. II, Gerald F. "Skimming Stories in Reel Time: An Experiment in Integrated Understsnding." Tech. Rep. 158, Yale University. Computer Science Dept. 1979.

(3) Dyer, Michael G. and Wendy G. Lehnert. "Organisation and Search Processes for Narratives/' Tech. Rep. 175, Yale Univ. Computer Science Dept. April 1960.

[4] Dyer, Michael G. In-Depth Understanding: A Computer Model of Memory for Narrative Comprehension.. PhD Thesis, Computer Science Dept. Yale University, (Forthcoming).

[5] Lehnert, Wendy G, Dyer, Michsel G., Johnson, Peter N., Yang, C. J., snd Steve Harley. "BORIS — An Experiment in In-Depth Understanding of Narratives." Tech. Rep. 188, Yale Univ. Computer Science Dept., Dec. 1980.

[6] Lehnert, Wendy G. The Process of Question Answering. Lawrence Erlbaum, NJ. 1976.

(7) Schank, Roger and Abelson, Robert. Scripts, Plans, Goals, snd Understanding. Lawrence Earlbaum, Hillsdale, NJ. 1977.

(6] Schank, Roger C. "Reminding and Memory Organisation: An Introduction to MOPs." In Strategies for Natural Language Processing. W. Lehnert and M. Ringle, editors, Lawrence Erlbaum, NJ (in press), 1981.

(91 Wilsosky, Robert. "Understanding Goal-Based Stories." Tech. Rev. 140, Yale Universally. Computer Science Dept. 1976.