ILP for Automated Telephony

Filip Zelezny[1], Petr Miksovsky[1], Olga Stepankova[1], Jiri Zidek[2]

[1] The Gerstner Laboratory for Decision Making and Control
Department of Cybernetics, Faculty of Electrical Engineering
Czech Technical University
Technická 2, CZ 166 27, Prague 6
{zelezny, miksovsp, step}@labe.felk.cvut.cz

[2] Atlantis Telecom s.r.o.
Žirovnická 2389, CZ 106 00, Prague 10
zidek@atlantis.cz

Abstract. We present an application of ILP in the domain of computerized telephony. Real-life data containing the logs of all calls in a company in different time extents are analyzed and rules are discovered that may help optimize the operation of the telephone exchange. We use the ILP system Progol to find a rule that identifies the origin of a transferred call. This rule is confirmed to have a very high precision and is then used to extract pairs of original/transferred calls in order to discover typical transferring schemes in the company, again by means of ILP. Some interesting schemes are found and are all graded and commented by a telecommunication expert who provided the source data.

1. Introduction

In spite of the explosion of information technology based on written communication, the most common and most frequently used tool is a telephone. Modern telephone exchanges provide comfort and effectiveness in managing telephone traffic. Everything works well as long as the basic condition is valid: the calling person knows where to find the person with whom he/she wants to communicate. If somebody tries to call you and you are not available, in a well-shaped company there is always a well-informed secretary, receptionist, operator or a colleague that can help give the desired information. In fact, the operator is a person with no direct product but with a very high impact on productivity of the other people. On the other hand, a wide range of companies cancel the post of the telephone operator – why?

- While reducing fixed costs, people that don't bring profit directly are lost first.
- It is not easy to find a person who is intelligent enough to be a good operator and to be modest enough to be only an operator.

This opens area for computer – the machine is paid only once so no fix costs set in. Moreover, the machine can work around the clock and can report detailed data suitable for analysis enabling improvements of the company organization.

Currently there are several domains where computers are used in private branch exchange (PBX) area (neglecting the fact that PBX itself is a kind of computer):

- Automated attendant – a device that welcomes a caller in a unified manner and allows him to reach a person, usually by choosing a person from a spoken list.
- Voice mail – a device allowing to leave a spoken message to a person not reached. Some sophisticated methods how to deliver the messages have been designed.
- Information service – the machine substitutes the man in providing some basic information; usually organized in an information tree.

In all these cases the calling party is required to co-operate.

The aim of the above listed tools is to satisfy a caller even if there is no human service available at the moment. But all devices are designed in a static, simple manner – they always act the same way. The reason is very simple – they do not consider who is calling nor what he or she usually wants – as opposed to a human operator.

By comparing a human operator or receptionist to a computer, we can imagine the following improvements of applications:

- Considering who is calling (by the calling party number) and what number is called, the system can learn alternatives which are most likely and allow the caller to access them in a more straightforward way or choose some probable alternative; knowledge can be obtained either from previous cases (taking into account other data such as daytime, explicit information – long absence of some employee, etc.) or by ”observing” the way how the caller was handled by humans before; this could shorten the caller’s way to get the information he/she needs.
- The caller can be informed by a machine in a spoken language about the state of the call and suggested most likely alternatives; messages should be “context – sensitive”

Naturally, the finite goal of a computerized telephony is a fully “duplex” machine that can both speak and comprehend spoken language so that the feedback with the caller can proceed in a natural dialog. See [4] for more overview on computerized telephony.

The starting point of an intelligent telephone exchange system lies in analyzing the history of calls to find out what actions can be simplified. For example, it may be discovered that it is always the case that if the external number E calls line L_1, it is always transferred to line L_2. Then it may be useful to redirect E to L_2 automatically, without the assistance of the person on L_1. Such hypothesis can be formulated in the language of Prolog in a simple manner, e.g. by using a predicate phone/4 with arguments (Calling_line, Called_line, Start_time, End_time):

```
phone(e,l2,T1,T2) :- phone(e,l1,T0,T1).
```

Finding such knowledge is a typical challenge for ILP namely due to the

- need for recursive notation of rules
- usage of variables
- implementation of domain dependent background knowledge.

While the first two reasons are clear from the above-listed clause, the third may become relevant e.g. in the case when the mentioned rule of redirection holds only in the peak hours defined by peak/1 in the background knowledge. Then the rule would read:

```
phone(e,l2,T1,T2) :- phone(e,l1,T0,T1), peak(T0).
```

Obviously, there is a wide scope of background knowledge applicable in the domain of telephone exchanges in companies. It may regard regular presence or absence of persons, special dedications of numbers (receptionists, fax/dial-up receivers, etc).

The experiments were initiated by having obtained real-life data from a telecommunication company describing all call exchanges in an anonymous company, taking place in 16 subsequent days including weekends. The resulting database contained about 6.000 records out of which about 7% were labeled as redirected (that is, being transferred from another call). For reasons explained in Section 2, for a redirected call it couldn't be directly identified from which call the redirection was initiated. This gave the reason for the first ILP task: find a precise and reasonable relation between the redirecting and redirected calls. After completing this task, we obtained two new datasets (from a different workplace) covering 3 days (1.800 records, 4% of redirected calls) and 31 days (11.000 records, 13% of redirected calls) respectively. The result of the first task - a rule describing the relation of redirection was tested on both of the new datasets and was confirmed to be very precise. The experiment is reviewed in Section 2.

Only on the basis of the first task's result we could proceed to search for typical redirection schemes, i.e. we aimed for schemes that appear on regular time basis or are results of some conditions expressed by means of background knowledge. The experiment was done with two of the available datasets and yielded some interesting rules. We review our ILP based search in Section 3.

All of the rules discovered within the two mentioned experiments were submitted for evaluation to a telecommunication expert from the data-providing company. The grading was based on intuitive interestingness (novelty), reliability, and usefulness. The expert also attached verbal comments on some of our results. His evaluation is reported in Section 4.

In Section 5, we review the directions at which our future collaboration will aim.

2. Identifying Redirections

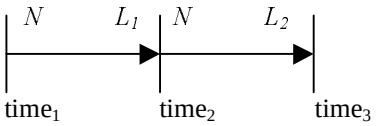
Each call is described by 11 attributes in the dataset: 1) date of call, 2) starting time, 3) ending time, 4) first party number, 5) second party number, 6) call type (*e* = incoming, *s* = outgoing), 7) setup mode (*a* = redirected, *d* = direct), 8) mode of

termination (*lb* = terminated, *li* = redirected to another line), 9) reason of termination (*rl* = local hang-up, *rd* = distant hang-up), 10) extension dialed by the calling party, 11) answering port of the exchange.

Party numbers are either 2 digits long for local lines or longer for external lines. Attribute 4) always identifies the local monitored party (it is the called line if call type = *e*), while attribute 5) may be local or external (it is the calling line if call type = *e*). Attribute 5) may also be blank if an external line is unidentifiable.

During the analysis, it turned out that only attributes 1) – 7) are reliable, while 8) – 11) are noisy or often missing due to malfunction of the exchange server logging. This was confirmed in communication with the data provider. Besides that, some call logs are missing completely if the exchange server does not have sufficient time to store them.

Although all calls originating from a redirection were properly labeled in the datasets (the setup attribute was *a* for redirected or *d* for direct calls), there is no pointer attached which would determine the original call. Our intuitive idea was that for every incoming redirected call, there must exist another call with the same calling party ending at the moment the new call starts (we couldn't judge on the basis of the attribute 8 due to its noisy character). The connection of calls matching this rule can be graphically shown as



where arrows denote a call from number N to a local number L_x.

Surprisingly, this rule (call it *naive_rule*) has only about 14% of precision and covers only 27% of redirected calls. In probabilistic notation[1], while $P(a) = 7\%$ it holds $P(a \mid naive_rule) = 14\%$ and $P(naive_rule \mid a) = 27\%$, where *a* stands for the event of call being redirected.

The *precision* measure $P(a \mid rule)$ as well as the *recall* measure $P(rule \mid a)$, which are in information retrieval defined equivalently as $TP / (TP + FP)$ and $TP / (TP + FN)$, respectively, are chosen here for rule evaluation because the rules will be used to extract examples for the experiment in Section 3; the former measure must be maximized to ensure that extracted positive examples will be correct and the latter measure should be maximized so that the set of positive examples in not too small. See e.g. [5] for a systemization of rule evaluation measures.

Due to the small precision of the *naive_rule* we looked for a better description with the ILP system Progol. We presented the *a* – labeled incoming calls as positive and *d* – labeled incoming calls as negative examples. The whole set of calls was presented as background knowledge by means of the predicate cll/9, together with predicates

[1] The *probability* $P(X)$ denotes here and afterwards the fraction of the cardinality of the set of records with the property X to the whole non-empty dataset cardinality. The *conditional probability* $P(X|Y)$ is defined here as the fraction of the number of records satisfying both X and Y to the number of records satisfying Y whenever $P(Y)>0$. A name of a *rule* may be used in place of a property X or Y, it then stands for the property of *satisfying* that rule.

distinguishing external and internal lines. Since the search with all available examples would have been computationally unfeasible, we decided to separate the data by days, perform the learning task for each day, and take the common subset of all resulting hypotheses.

Each positive example was of the form

```
rcall([09,08,55], [09,09,49], 50, 0281007, e, lb, rd,
empty_num, 005040).
```

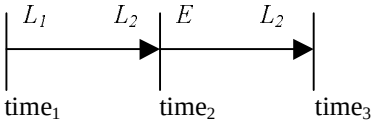
The date attribute was omitted due to its irrelevance in the one-day-domain learning. The setup mode attribute was obviously omitted in the examples, as we needed a description that does not use this attribute. All constants beginning with the string "empty" denote missing attribute values. As a result, we obtained the following rule R_1:

```
rcall(A,B,C,D,E,F,G,H,I) :- cll(J,A,C,K,E,L,M,H,N),
external(D), local(K).
```

Verbally, this rule says that a call with an external party has been redirected if there is another call which:

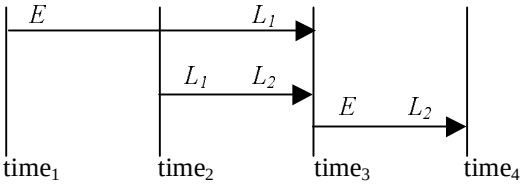
1. has the same local party number as the redirected call (C variable),
2. was released when the redirected call started (A variable),
3. its second party number is local,
4. it has the same call type (i.e. it is also incoming) and 'extension dialed' attributes as the redirected call (E, H variables).

Rule R_1 can be described by the following graph of subsequent calls:



(where L stands for local and E for external lines).

Although this rule contradicted with our intuitive expectation, its precision on the whole 16-day domain was $P(a \mid R_1) = 99\%$ and it covered about 21% of all redirected calls ($P(R_1 \mid a) = 21\%$). After obtaining two new datasets, the rule was tested on both of them and again reached a very high precision and also good recall: $P(a \mid R_1) = 100\%$, $P(R_1 \mid a) = 34\%$ on the 3-day domain, $P(a \mid R_1) = 99\%$, $P(R_1 \mid a) = 54\%$ on the 31-day domain. Only later we learned that the scheme described by R_1 is a subgraph of a 'transfer with notification' scheme where L_1 calls L_2 to notify the person on L_2 about E being transferred, as can be seen bellow:



This scheme, however, could not be identified from the source data, as the exchange server failed to log the $E - L_1$ calls in the majority of cases.

We have thus found an accurate alternative rule identifying redirections in the noisy data domain.

3. Searching for Typical Redirections

Rule R_1 discovered by Progol has, besides its very high precision, another advantage: it contains all the information about a redirection: the calling party (E), the redirecting party (L_1) and the destination party (L_2). With the knowledge contained in R_1 we extracted the set of redirections in the form of positive ground facts of the form

```
redir([04], [09,24,24], 81940398, 22, 26, e, d,
empty_num).
```

This example describes a redirection that took place on day[2] 4, when the external number 81940398 called line 22 at 9:24:24 and was later redirected to line 26. The three last arguments (*e* for incoming call, *d* for direct set-up, missing extension) are all related to the first (81940398 – 22) call. We do not introduce attributes related to the call initiated from redirection nor the time of redirection. The reason is that we are interested in finding a rule determining the conditions when a call coming to a local line should be redirected automatically to another line (e.g. because such a redirection always took place in the past). At the moment of application of such a rule, the attributes of the following call are not known; therefore, utilizing them in the rule would not make sense.

As negative examples, we can use all calls that were released without being transferred to any other line. Specifically, those calls which ended at a moment when no other call initiated – these calls were for sure not redirected to another extension. Progol offers an elegant way to describe these examples, allowing non-ground facts as examples (see [1] or [2] for details on Progol). For instance, the family of negative examples, resulting from a single non-transferred call from 44091118 to 31 on day 10, time 10:12:18, has the form:

```
:-redir([10], [10,12,18], 44091118, 31, X, e, d,
empty_num).
```

[2] The list-notation of days (such as '[04]') was used to distinguish between the integer type of extension numbers and day numbers as they overlap.

meaning that on day 10, time 10:12:18, 44091118 called 31 and was not redirected to any line (X is a variable here).

Since we wish to have rules saying under which conditions a call should be redirected to a certain line, we instruct Progol to form rules with the head given by the declaration

```
:-modeh(1,
redir(+date,+time,+external,+local,#local,+type,+setup,
+local))?
```

i.e. the destination line should be identified by a constant.

The background knowledge definition consisted of predicates defining special daytimes (e.g. morning and afternoon peaks figured out by a distribution graph), equivalence of weekdays (Mondays, Tuesdays, etc.), fax/collect/shared/reception numbers, different kinds of answering ports etc. It consisted of about 60 lines of code.

On the first dataset (16 days), with 118 positive examples and about 2.000 negative examples, we obtained the following 7 rules (besides singularities not covered by the rules) which can be divided into three groups:

Group I.

```
redir(A,B,C,D,55,E,F,G) :- number_is(D,17),
setup_is(F,a).
```

(covers 20 examples)

```
redir(A,B,C,D,10,E,F,G) :- number_is(D,17),
setup_is(F,a).
```

(covers 5 examples)

```
redir(A,B,C,D,35,E,F,G) :- number_is(D,17),
setup_is(F,a).
```

(covers 9 examples)

This group of three rules says that if 17 is called and this call has already been redirected, there will be another redirection onto 55 or 10 or 35. The conclusion of the combination of the three rules is not unique; we only have a probability estimation given by the fraction 21 / 6 / 10. We shall later show how to avoid this ambiguity.

Group II.

```
redir(A,B,C,D,16,E,F,G) :- number_is(D,16),
type_is(E,s).
```

(covers 38 examples)

This is a ‘strange’ rule; it says that line 16 calls out (s for outgoing call type) and then redirects the call to itself (it happens 39 times in the 118 examples).

Group III.

```
redir(A,B,C,D,55,E,F,D).
```

(covers 11 examples)

```
redir(A,B,C,D,10,E,F,D).
```

(covers 4 examples)

```
redir(A,B,C,D,35,E,F,D).
```

(covers 6 examples)

This group of rules says that if the dialed extension equals[3] the number of the local party, the call will be transferred to 55 or 10 or 35. Again, the result is ambiguous and we have the fraction 12 / 5 / 7 as a probability estimation.

We may use another feature of Progol to avoid the ambiguity of the resulting rules. This can be done by a constraint declaration:

```
:-redir(A,B,C,D,X,F,G,H), redir(A,B,C,D,Y,F,G,H),
different(X,Y).

different(X,Y):-X>Y.

different(X,Y):-Y>X.
```

i.e. we require that resulting rules with the same 'input' arguments may not produce different 'output' constant. Under such conditions, the result consists of the rule in Group II and another rule R_2:

```
redir(A,B,C,D,55,E,F,D) :- number_is(D,10).
```

(covers 2 examples)

which means that a call to 10 is transferred to 55 if the dialed extension was 10. Although the result is now rather modest, it contains two non-ambiguous rules.

On the second dataset (3 days), a similar experiment was not performed due to the small extent of relevant examples.

On the third dataset (31 days), with about 480 positive and 6.000 negative examples, we obtained the following rule R3 :

```
redir(A,B,C,D,11,E,F,G) :- number_is(C,61000000),
number_is(D,31), day_is(A,tuesday).
```

(covers 4 examples)

which says that every Tuesday, when 61000000 calls[4] 31, it is transferred to 11. It should be realized that due to the presence of both positives and negatives from the

[3] Although this equality is rather natural, the information is contained in the fact that the dialed extension attribute is not missing. In the majority of records it was blank.

whole extent of the 31 days, it was never the case that 61000000 called 31 on Tuesday without being transferred. Also, we checked in the database that these redirections did not concentrate in one Tuesday.

4. A Domain-Expert's Evaluation and Comments on Discovered Rules

We submitted all of the discovered rules to the data-provider together with a grading table. The grading is based on intuitive interestingness (novelty), reliability, and usefulness. The grading range was given as follows:

Interestingness
0 – Rule is trivial or this fact is known to me (not interesting).
1 – New knowledge.

Reliability
0 – Rule is intuitively unreliable, could be a result of wrong usage or operation of the exchange or wrong learning setting.
1 – Rule is intuitively reliable, no contradiction with my expectation.

Usefulness
0 – This type of rule has little chance of being applied.
1 – This type of rule has good chance for a real-life application.

All that follows until the end of the section is the expert's grading and his comments.

RULE	In Text	I	R	U
characterization of redirections **valid on: all data** (A,B,C,D,E,F,G,H,I) :- ticket(J,A,C,K,E,L,M,N,H,O), external(D), local(K).	R_1	0.5	1	1
redirections from 17 (ambiguous) **valid on: 1st dataset** redir(A,B,C,D,55,E,F,G) :- number_is(D,17), setup_is(F,a). redir(A,B,C,D,10,E,F,G) :- number_is(D,17), setup_is(F,a). redir(A,B,C,D,35,E,F,G) :- number_is(D,17), setup_is(F,a).	*Gr. I*	1	1	1
self-redirections on 16 ("strange rule") **valid on: 1st dataset** redir(A,B,C,D,16,E,F,G) :- number_is(D,16), type_is(E,s).	*Gr. II*	1	0	0
redirections when local no.= dialled ext. (ambiguous) **valid on: 1st dataset** redir(A,B,C,D,55,E,F,D). redir(A,B,C,D,10,E,F,D). redir(A,B,C,D,35,E,F,D).	*Gr. III*	1	0	0
redirections from 10 when local no.= dialled ext. **valid on: 1st dataset** redir(A,B,C,D,55,E,F,D) :- number_is(D,10).	R_2	0	1	0

[4] The 'strangeness' of the number 61000000 is not important, it will be commented in Section 4.

Tuesday redirections **valid on: 3rd dataset** redir(A,B,C,D,11,E,F,G) :- number_is(C,61000000), number_is(D,31), day_is(A,tuesday).	R_3	1	1	1

Comments:

R_1: This is a crucial rule; it should be used as background knowledge.

Gr. I: 17 is a laboratory where redirections are actually frequent; 35 is inventory, 10 is customer service, 55 is a shared number.

Gr. II: I have no idea how this could happen.

R_2: Expected rule: this redirection is 'wired' in the exchange.

R_3: This rule is certainly correct (and interesting): the secretary is called by her husband; her line is not directly reachable from external lines. This is exactly the type of rule we need to search. The zeros in the suffix of the external number are substituted by the internal exchange.

5. Concluding Remarks

We have shown how ILP can be applied in the domain of automated telephony. Our approach was incremental in the following sense.

First, a rule was discovered that accurately defined the relation of one call being transferred to another call. The precision of this rule was later confirmed on newly obtained data.

Second, using that rule, redirection pairs were extracted from the database and presented as positive examples in a search for typical redirection schemes. For specifying negative examples we took advantage of Progol allowing non-ground facts as examples.

As a result, we obtained several rules describing some typical redirections. These rules were submitted to the data-provider for grading and comments.

With respect to this evaluation, we have very optimistic expectations for future experiments, which will mainly focus on:

- extending background knowledge by more domain-specific information, such as regular presence/absence of persons, their role in the company, categories of business partners recognizable by their telephone numbers etc.
- tracing the behaviour of specific lines.
- attempting to implement discovered rules into the background knowledge for subsequent experiments.
- implementing the experiments into a theoretical framework for treating temporal data as shown e.g. in [6].

This paper is accompanied by a web site with all data necessary to review/repeat our experiments by the reader (e.g. as the raw inputs, the Progol input/output codes, complete listing of background knowledge, SQL queries needed for rule evaluation etc.). Moreover, some additional comments and links to relevant sites are attached.

The URL is *http://labe.felk.cvut.cz/~zelezny/ilptelephony.*

Our preliminary experiments prove that the telecommunication domain offers a wide scope for application of ILP, even outside of the area of classical data warehousing techniques described in [3].

6. Acknowledgements

The authors are grateful for the support by the following grants: CTU 30008803 (CTU internal grant), Project INCO 977102 ILPnet2 and IST-1999-11495 Sol-Eu-Net. The first author wishes to thank the ILP 2000 organizers for the subsidy on the registration fee.

The authors also wish to thank the anonymous referee for his/her helpful remarks.

7. References

[1] S. Muggleton: Inverse Entailment and Progol. New Generation Computing Journal, Vol. 13, pp. 245-286, 1995.

[2] S. Muggleton. CProgol4.4: Theory and Use. ftp://ftp.cs.york.ac.uk/pub/ML_GROUP/progol4.4/

[3] R. Mattison: Data Warehousing and Data Mining for Telecommunications, Artech House, 1997

[4] J. Zidek, O. Stepankova: Automated Intelligent Handling of Phone Calls in PBXs. http://www.matra.cz/ilp/slides.doc

[5] N. Lavrac, P. Flach, B. Zupan: Rule Evaluation Measures: A Unifying View. Proceedings of the 9th Inductive Logic Programming Workshop (ILP99), pp. 175-185.

[6] S. Moyle, S. Muggleton: Learning programs in the event calculus. Proceedings of the 7th Inductive Logic Programming Workshop (ILP97), pp. 205-212.