1 Introduction

The INCOM system aims at supporting novice programmers in the Prolog programming language in solving simple standard programming tasks by providing help in the event of an error made by the user. This report summarizes the results of an evaluation of the system. The INCOM system has been tested with respect to several conceivable common mistakes made by novice users and has been evaluated in terms of utility of the feedback returned to the user. The primary goal of this evaluation was to point out problems and bugs in order to improve the system before proceeding in the project.

2 Analysis description

In this evaluation the aspects that have been examined are:

- typographical errors in predicates, constants and variables
- confused or alternated argument positions
- omitted arguments in predicates
- additional arguments in predicates
- missing or superfluous subgoals
- incorrect linking between subgoals
- incorrect use of (in-)equations
- misuse of anonymous and standard variables or constants
- efficiency influenced by the order of subgoals
- semantically completely wrong queries

Two example domains, namely “Häuser” and “Bücher”, were used to test the system. In the appendix, a complete listing of applied queries, their feedbacks and a brief utility estimation is given.

3 Analysis results

The evaluation shows that for several types of errors the INCOM system gives appropriate feedback. In detail that means that the user is shown the source of his error, given an explanation in which way it does not work and a hint how to correct the query. Still a complete solution is not to be given as this would be of little pedagogical value. Instead the user is asked to reconsider the way he tried to solve a task and find the way to the solution himself and hence extending his knowledge about the Prolog language.

In several cases the system succeeds in giving useful feedback. However there are as well types of errors which the system does not give reasonable feedback to and in some cases could even confuse a beginner. Apart from this a few bugs have been discovered. For examples and further explanation, see the appendix[1].

[1]Links to test queries are of the following form: SiQt = **S**ection in Appendix - **i**ndex number of task - **Q**uality of feedback (**P**ositive, **N**egative, **U**nsatisfactory) - test case number. B2Nb means appendix B, task 2, negative example b

3.1 Strengths

The INCOM system is particularly successful when dealing with the following types of mistakes:

- a superfluous subgoal has been added to an otherwise fairly correct query:
  The system encourages the user to reconsider whether that subgoal is necessary

- instead of a constant given in the task a variable has been used:
  The system proposes looking up the correct constant in the task

- exactly one letter of a predicate name has been misspelled:
  The system marks the mistake and proposes the correct predicate name

- in an act of alternating subgoal positions a variable used in an (in-)equation is not bound when evaluated:
  The system mentions that and hence encourages the user to correct the predicate order

- argument positions have been alternated which produces type conflicts:
  The type conflict is marked drawing the user's attention towards the mistake

- an incorrect constant which is used in other ground terms has been used instead of the correct constant:
  The user is encouraged to check the wrong constant and apply the correct one given in the task

- argument positons of a variable and the correct constant have been confused:
  The system prompts that the constant is at the wrong position

- the user applied a constant instead of an anonymous variable:
  The system asks whether any information in this slot is needed and proposes discarding the constant by replacing it by an anonymous variable

- syntactical errors (see exceptions below):
  Barring there are too many syntax errors the system informs which symbol is missing and sometimes even where approximately it is supposed to be. Sometimes however, the mistake could be localized more precisely in order to ease the search for it

3.2 Aggravating properties

Types of errors for which the system compromises the user's advance and understanding are as follows:

- instead of an anonymous variable a standard one has been applied:
  The user is adviced to use a constant that can reputedly be found in the task (A1, B1Na, B2Nb, B3Uh, B3Nd, C3Na)

- a constant has been misspelled so that the entered one does not appear in any ground term:
  The system asks whether the user meant to use a (standard) variable and recommends writing it with a capital letter. Even single faults in spelling are not detected (B1Nb,C2Na)

- a required inequation has been replaced by an equation:
  The equation is not considered to be linked to the correct inequation and therefore marked as superfluous while a missing subgoal is mentioned. In addition, the system expresses doubts whether constants used in the equation are given in the task. One time, the system got into an infinite loop not giving any feedback at all (A2, B2Na, C4Ua)

- instead of a variable an anonymous one has been used:
  The system inconsistently proposes applying a constant and in another error message using a variable. This contradiction probably decreases a user's confidence in the system (B2Nb, B3Nd, C1Na,C1Nb, C1Nc)

- inequations using the symbol `\=` or other ways of direct negation are impossible:
  *The system cannot process such expressions and commonly replies that too many syntax errors are in the query (B3**, B4**)*

- a variable linking one subgoal to another has been misspelled:
  The system is unable to recognize the connection between the subgoals. Again, at least a single fault correction is most desireable to overcome this problem (B2Pb, B3Nb, C1Uf, C5Ua)

- a constant given in the task has (maybe additionally) been used in place of a standard variable:
  The user is asked whether this constant is given in the task. As it is given indeed, a beginner could be irritated (B2Nc,C4Pa)

- a comma seperating two subgoals was incorrectly placed (e.g. `S_1() ,S_2()` instead of `S_1(), S_2()`):
  This typographical error causes the system to fail processing the query. A small slip like this should rather be detected as it is especially difficult and cumbersome to find an error like that (C4Na)

3.3 Insufficient properties

There are several situations in which the system does not provide a sufficient feedback, namely:

- a required subgoal has been left out:
  The error message is fairly complicated to understand and a novice user could fail to extract the meaning: Suggesting that another subgoal is needed. See examples for more details (B1Ua,B2Nc,B2Ua,C1Pa,C1Pd,C4Pb)

- the user is ignorant of the exact spelling of a constant needed when this is not explicitly given in the task:
  The system only tells the user to review the task but does not help to find out the spelling (B1Pb)

- the user did not fill all slots/argument positions of a subgoal:
  In some cases this problem is pointed out which is sufficiently helpful but sometimes the system seems to be unable to properly process the query calculating for a long time and returning an interface in disarray (see list of bugs) (A6,C3Pb)

- the user added extra arguments to a predicate:
  The system calculates for a long time (> 1 minute) and returns an interface in disarray (see list of bugs) (B1Ub,C1Ua)

- a wrong constant has been used in an inequation: *The user is advised to have a close look at the inequation again but in the error message the variable name is replaced by it's internal name like _G7399 which causes confusion (B2Ub)*

- a type conflict has been caused e.g. by alternating argument positions:
  Besides suggesting to solve the type conflict the system advises the user to look up some kind of help for this type of error. The system fails to give information where this help can be found (B2Uf,C1Pb)

- a syntactically correct but otherwise awkward query has been entered:
  In certain cases like this the system does not give any feedback at all. It appears to be unable to deal with this type of errors. (B2Ue,B4Ua,B4Ub,C1Ud)

- the query is semantically incorrect and as well contains syntax errors:
  The system gives suggestions how to correct the syntax errors but does not mention that the user is headed for the wrong direction in first place. It should rather dectect and point out obvious misconceptions prior to syntax errors (B3Nb)

- more than one spelling error occured within one predicate:
  The system asks the user to check the predicate names which is appropriate but the error message states that the error could not be identified which is bewildering (B1Ui)

- a query contains many different or connected errors:
  The user is faced with lots of error messages which is pedagogically not advisable (B3Nd, C5Ua)

- a query has been formed in a correct but inefficient way by altering the order of the subgoals:
  The query is considered to be correct and the user is not informed about more efficient ways to solve the task (C1Ub)

- a task has been solved in a correct but complicated way, e.g. instead of a constant in a predicate, a variable is used and bound in an equation afterwards:
  The system does not judge the query to be correct (C2Ua)

- a wrong constant has been used in a predicate:
  In some cases, using a variable is proposed as mentioned above. Otherwise the incorrect constant is highlighted but there is no further feedback (C1Ue)

- links between subgoals have been omitted which are implicitly required to solve the task:
  The system suggests to look up the required informatik in the task where it cannot be explicitly found (C5Ua)

3.4 Bugs

In the aforementioned sections it has already been alluded that the system is not complete and thus cannot adequately process all queries. In some cases, no feedback is given at all and the user is just prompted to try again. In one example the system got into a probably infinite loop. The most obvious bug, however, is that in certain cases the interface is in disarray after processing the query. In detail that means that where the diagnostics would be expected, there are extra drop down menus to choose a task and enter a query plus on the bottom of the page another section called "Diagnose" asking the user to choose a task. It appears that this problem emerges especially often when a predicate has too many or too few arguments supplied.

3.5 Highlighting

Looking through the examples in the appendix one can find several examples in which the highlighting does not work properly. The most common error is that faulty expressions are not highlighted but in some cases there are words highlighted which do not actually have much to do with the errors made. In general, it seems that additional or non-existant predicate names used in a query are not highlighted as well as variables used in faulty (in-)equations.

3.6 Improvement ideas

While testing the system three aspects have been identified which can be improved in order to enhance the performance of the INCOM system.

The first one is that in the current INCOM interface the user must not enter a dot at the end of the query or a syntax error will occur. This is not considered very useful in improving a beginner's skills as it might drive the user to forget the dot in future when working without the INCOM system.

Secondly for more complicated tasks like "Häuser-4" a single line to enter a query is hardly enough to solve the task in a conveniant way. Defining help predicates and generally being able to structure the queries is necessary for those tasks.

Thirdly the INCOM system does only offer static help. That means that if a user keeps making mistakes and is unable to complete the task with the standard help given by the INCOM system, he is likely to desperate and give up. To engage this problem a dynamically adjustable help is necessary that gives small cues on first tries and extends help on later errors if the user seems to be unable to successfully apply the general suggestions. Generating user profiles and keeping track of their development could be part of this feature in order to provide approriate help at any point.

4 Conclusion

The INCOM system is capable of aiding the user in solving simple programming tasks in a restricted range of possible mistakes. The combination of highlighting expressions in the task description as well as in the users faulty query and giving more detailed error messages can be regarded quite useful. However, the system lacks capability to process certain types of common mistakes and is not complete. Some errors are misinterpreted which entails that the user gets inappropriate confusing or even encumbering feedback. Out of these reasons, the current version of the system is not adequate to aid novice programmers. A user will most probably not be patient enough to wait for the system to generate feedback which fairly possibly will not be helpful.

The main points that have to be worked on are the following:

- use of constants, anonymous and standard variables
- detecting syntax errors concerning the number of arguments in predicates
- error detection for equations and inequations
- dealing with spelling errors in variables or constants
- negation
- errors in linking of subgoals
- dealing with unusual and over-complicated ways of solutions
- incremental help

APPENDIX

evaluation notes and specific queries

A General notes

1. When using standard variables in queries where anonymous variables would be adequate the given feedback is 'An dieser Stelle sollte es eine Konstante sein. Finde diese Angabe in der Aufgabestellung.' This feedback could irritate a user as a constant would do even worse from the semantic point of view. Furthermore, the information to correct the statement is not given in the task description hence probably causing further confusion.

2. Misspelled constants which are not used in any ground instance of a certain predicate are assumed to be variables in the hint messages. A user misspelling a constant in a query when a constant is required is thus lead astray.

3. In case that the system is unable to process a possibly syntacitically correct query and in some cases maybe times out, the web interface is contorted. There are two sections for choosing a task and entering queries.

4. The fact that the user must not enter the dot at the end of his query compels that the user gets accustomed to this at first and might cause the user to forget it when programming without the INCOM system in future.

5. In the examined version of the INCOM system all types of help are constant in terms of error repetition by the user. A gradually increasing level of detail in help would be desireable.

6. When entering too few terms in a predicate / not filling a sufficient amount of slots, the system is rather unpredictable whether it detects the error and outlines that more slots have to be filled or whether it takes a long time of calculation and just returns a contorted interface.

B Domain Haeuser

1. In welchen Strassen gibt es Mehrfamilienhaeuser?
   correct query: obj(_,mfh,Strassenname,_,_).

Positive examples

(a) obj(On,mfh,SName,Hnr,_), bew(_,On,_,_,_,_).
The user has included a superfluous predicate in his query.
the superfluous term bew(...) is not highlighted in the diagnostic section. However, the help text 'Brauchst Du diesen Term? Es scheint überflüssig zu sein' might encourage the user to reconsider it's meaning and probably drives him to omit the term on next try. The highlighting should be extended to this case.

(b) obj(_,Typ,Sname,_,_).
The user was unable to map the meaning of 'Mehrfamilienhäuser' into the proper constant needed in the correct query. Among others, one reason could be that he is ignorant of the exact spelling of the required constant.
The help saying 'An dieser Stelle sollte es eine Konstante sein. Finde diese Angabe in der Aufgabestellung.' precisely covers the problem and thus should give the user an idea of where to search for his mistake. However, if the user does not know which constant to apply ('mfh' in this case) this help will not ease his problem.
A possibility to list possible constants that could be used in the particular slot is desireable.

(c) obj(_,efh,Sname,_,_)
The user has used the wrong constant.
In the error term the incorrect constant is highlighted and the general answer advices the user to reconsider that he is supposed to find 'Mehrfamilienhäuser'. This is probably a highly suitable hint allowing the user to concentrate on his mistake while not giving the solution.

(d) obj(o1,efh,gaertnerstr,15,1965)
The user, maybe completely confused by the task, has just copied the example fact in the database.
The error messages comprise three instances of incorrectly used constants and on instance of the inappropriate use of the constant 'gaertnerstr'. The hints given state that in the first three cases an anonymous variable would fit and instead of 'gaertnerstr' a Variable should be used. Apart from that, like in the previous example, the constant 'efh' is marked inappropriate.
The help to the user's mistakes gives almost the full solution to the first problems. Giving the user only the explanation stating 'Diese Information brauchst Du nicht, oder?' respectively 'War diese Information in der Aufgabe gegeben?' could be sufficient to guide the user to the correct solution.

(e) obk(_,mfh,Str,_,_)
The user misspelled the name of a predicate.

The answer stating that the predicate ‘obk’ is non-existant and suggesting the correct name instead probably greatly helps the user. The slight mistake can be corrected without inconveniant searching.
Extending the detection of spelling errors is generally preferable.

(f) obj(mfh,Strname)
The user omitted all unnecessary slots in the predicate.
The error message reads ‘1.Teilziel Dieses Teilziel erwartet mehr Argumente Prüfe die Stelligkeit des Teilziels nach.’ and thus lets the user know the type of his mistake. This help should be sufficient though the error message is printed out twice.
Unfortunately the system does not always react this way. See A6.

Negative examples

(a) obj(On,mfh,SName,Hnr,Bj)
The user has used non-anonymous variables in all slots and would therefore get too many details in an answer.
→ general 1)

(b) obj(_,mfg,Sname,_,_)
The user misspelled the constant ‘mfh’.
The answer ‘Soll das eine Variable sein? dann muss der erste Buchstabe gross geschrieben werden.’ distracts the user from the real problem and could enforce a novice user to change his query in a way that leads further away from the correct solution.
When a constant is required by the task, the system should refer suggest revising the constant used in the query and checking for spelling errors. See also B1Pc.

Unsatisfactory examples

(a) bew(Vnr,On,_,Kaeufer,_,_).
The user has left out the correct predicate and tried to solve the task with another one.
The term bew is marked is useless, but the other error mentioned, saying ‘0. Teilziel Dies reicht noch nicht. Bitte prüfe, wir müssen mehr Teilziele haben.’, is most probably not very useful to a novice user. I can only assume that it is supposed to mean that a predicate capable of fulfilling the task is not included in the user’s query.
The error message should be clarified.

(b) obj(_,mfh,Snr,_,_,T)
The user invented another argument.
The system takes a very long time to calculate before returning an interface in disarray without any feedback.
The rather simple problem of applying too many arguments should be detected.

2. Welche Häuser (Objektnummer) sind vor 1970 erbaut worden?
correct query: obj(Onr,_,_,_,Bj), Bj <1970

Positive examples

(a) Bj < 1970, obj(Objnr,_,_,_,Bj)
The user has alternated the two subgoals.
The error description states 'Bj Diese Variable ist noch nicht instanziiert. Diese Variable sollte instanziiert sein, bevor sie für eine arithmetische Operation verwendet wird' leading the user's attention to the actual problem. In addition the user is reminded of the rules of using arithmetical statements which gives this diagnose a high pedagogical value. The diagnose fails to highlight the variable 'Bj' in the answer message, however.
The highlighting should be improved to cover this case.

(b) obj(Objnr,_,_,_,Bj), Baujahr < 1970
The user has applied two different variables where the same would be appropriate.
Like in the previous example, the error message states that 'Baujahr' is not instantiated yet which tells the user that something is wrong with this variable and might lead him to understanding his mistake. Nothing is highlighted in the answer message.
It would be nice if the error message linked the two variables. Correct highlighting.

(c) obj(Onr,_,_,_,<1970)
Another way to try to solve the task with one subgoal only.
The diagnostics state that there are too many syntactical errors in the query, probably reminding a user that such a construction is not possible.

Negative examples

(a) obj(Onr,_,_,_,Bj), Bj = 1970
The user has misinterpreted the task or misspelled the symbol '<'.
The answer highlights the words 'vor' and 'erbaut'. This is suitable in this context. The explanation and help says 1) '1. Teilziel Dies reicht noch nicht. Bitte prüfe, wir müssen mehr Teilziele haben.' and 2) '1970 War diese Information in der Aufgabe gegeben? An dieser Stelle sollte es eine Variable sein.' 1: In this case of an erroneous query this error message is confusing as the user actually applied the correct number of subgoals and only made a slight mistake in the second one. 2: This help can easily lead to frustration as the term '1970' is given in the task description and a variable would be incorrect.
The system should ask whether the user really meant to use an equation instead of an inequation in this task. Dealing with other matters like the number 1970 in this case could be neglected if it cannot be handled appropriately.

(b) obj(Onr,_,_,Jahr,_),Jahr < 1970
The argument position for 'Jahr' was alternated.
The answer message highlightes the two confused variables ('Jahr'

and '_8620') which should easily allow the user to notice the mistake. The error message though consists of three lines: Following it, the variable 'Jahr' is supposed to be a constant (→ A1)) and - which is most confusing - '_8620' should be a variable AND a constant simultaneously: '(_8620) War diese Information in der Aufgabe gegeben? An dieser Stelle sollte es eine Variable sein.' - '_8620 An dieser Stelle sollte es eine Konstante sein. Finde diese Angabe in der Aufgabestellung.' This obvious contradiction decreases the user's trust in the system.
Rules suggesting the use of constants or (anonymous/standart) variables should be refined.

(c) obj(Onr,_,_,_,1970)
The user tried to integrate the date into the first subgoal directly.
The number '1970' as well as the word 'vor' in the task description is highlighted, possibly allowing the user to correct the mistake. The error term states that a subgoal is missing but also says that the constant '1970' is not given in the task. A variable should be applied here. The fact that a variable is necessary is correct but questioning whether '1970' is given in the task could cause frustration.
A check whether any constant used in a query is relevant to the task at some place before assuming it is not given in the task would help here.

Unsatisfactory examples

(a) obj(Onr,_,_,_,Bj)
The user forgot to add a second subgoal constraining Bj.
The answer highlights the word 'vor' and thus gives a hint that can help the user become aware of the missing subgoal. The explanation and help stating '1. Teilziel Dies reicht noch nicht. Bitte prüfe, wir müssen mehr Teilziele haben.' is rather inexpressive, however. A user not familiar with the system internals will most probably have difficulties to link the expression '1. Teilziel' to the actual task.
See B1Ua.

(b) obj(Objnr,_,_,_,Bj), Bj < 1980
The user applied the wrong number in the inequation.
The answer message does not highlight anything though in this case highlighting the wrong number '1980' would be most desireable. The error message '_G7368 Schau dir die Ungleichung noch einmal ganz genau an.' gives information about the location of the error but the expression '_G7368' can cause confusion.
Improve the highlighting - otherwise using the original variable name is recommended.

(c) obj(Objnr,_,_,_,Bj), Objnr < 1970
The wrong variable was used in the inequation.
The answer message does not highlight anything. The error description states 'Objnr Diese Variable kann nicht innerhalb eines Ziels gleichzeitig mehrere Typen haben Siehe typischen Typkonfliktsfehler [Objnr=number, Objnr=objektNr]' and thus helps locate the error

and gives a hint about the error type. However, the help message 'siehe typischen Typkonfliktsfehler...' suggests looking up the type of error somewhere but leaves the user puzzled about where this information can be found.
Apart from highlighting, the error message has to be refined.

(d) obj(Objnr,_,_,_,Bj), Objnr > o1
The wrong variable together with a possible ground term was used in the inequation - possibly as a try to correct the previous mistake by going into the wrong direction.
There is no diagnostics at all and no error message. A sign of incompleteness in the system.
The inequation could be identified as non relevant to the task in this context.

(e) obj(Objnr,_,_,_,)
The user omitted the information about the date of construction and the adherent inequation.
The error description is empty and the answer message does not highlight anything.
A hint concerning the date being required in the task would be most helpful.

(f) bew(_,Objnr,_,_,_,Dat), Dat < 1970
The user misunderstood the task and uses the wrong predicate.
Nothing is highlighted and the error message reads 'Dat Diese Variable kann nicht innerhalb eines Ziels gleichzeitig mehrere Typen haben Siehe typischen Typkonfliktsfehler [Dat=number, Dat=datum]'. The syntax error is detected but this could drive the user to "correct" the query like in the following example, still ignorant of the semantical error that he is asking for the wrong facts.
Use of the wrong predicate should be detected and dealt with prior to syntax errors.

(g) bew(_,Objnr,_,_,_,Dat), Dat = d(Jahr,_,_), Jahr < 1970
Maybe having made the previous mistake, the user tried to correct it this way.
The answer message highlights the word 'Häuser' while the error message gives two errors: First the slightly confusing explanation '1. Teilziel Dies reicht noch nicht. Bitte prüfe, wir müssen mehr Teilziele haben.' and secondly 'bew(...) Brauchst Du diesen Term? Es scheint überflüssig zu sein.'. This might give the user an idea that the predicate 'bew' is inappropriate for this task.
Explicitly mentioning the semantical error would be nice.

(h) obj(Onr,Typ,_,_,Jahr),Jahr < 1970
In addition to the actual task, the user asks for further information ('Typ').
The answer message highlights the inappropriate variable 'Typ' but the error message proposes a constant. (see A1))

(i) ovh(Onr,Typ,_,_,Jahr),Jahr < 1970
The predicate name is misspelled.
The error message reads 'Fehler kann nicht identifiziert werden. Deine

Anfrage passt nicht zur Datenbank. Bitte prüfe die Prädikate nach' which is presumably sufficiently helpful. The message that the error could not be indentified is a bit bewildering.
Leaving out that the error could not be identified would solve the problem.

3. Auf welchen Grundstücken stehen keine Mehrfamilienhäuser?
correct query: obj(_,Typ,Str,Hnr,_), Typ m̄fh.
This in NOT judged correct by the INCOM system!

Positive examples

(a) obj(_,efh,Str,Hnr,_)
Assuming that there are only base facts with the constants either 'efh' or 'mfh', the query is solved this way - of debatebal correctness. The answer message highlights 'efh' and 'keine Mehrfamilienhäuser' which is an adequate notion of the mistake. A user is forced to reconsider whether his current solution meets with the required criteria. The error terms mention the missing subgoal thus supporting the user in finding the correct solution. Secondly the error message questions the constant 'efh' and proposes a variable instead.

(b) obj(_Typ,Str,Hnr,_), Typ m̄fh
A comma is missing between _ and Typ.
The error message states that a comma is missing and that this produces a syntactical error. In larger queries than this one a user would still be forced to search through several lines of code looking for the missing symbol.
Mentioning in which subgoal or between which the syntax error occured would be helpful.

Negative examples

(a) obj(_,Typ,Str,Hnr,_), Typ m̄fh
The correct query.
The answer message is that there are too many syntactical errors.
Processing negation correctly is inevitable for such tasks.

(b) obj(_,Type,Str,Hnr,_), Typ = mfh
A slight spelling error in the variable 'Type'.
The feedback given criticizes a missing subgoal while the task description has the word 'keine' ('Mehrfamilienhäuser?') highlighted. While the highlighting could refer to the semantical error in using an equation ('Typ = mfh') instead of an inequation, the user's attention is driven away from this matter by the error message. As there is no subgoal missing but there rather is just a typographical error, a novice user will have difficulties to correct the query.
Detecting a spelling error like it is done with predicate names would be quite helpful.

(c) obj(_,Typ,Str,Hnr,_), Typ = mfh
The above spelling mistake is correct, the semantical error remains. The task description highlights ‘keine Mehrfamilienhäuser’, the answer message marks the variable ‘Typ’ within the goal ‘obj(...)’ and the error message states that there is a subgoal missing plus secondly ‘mfh War diese Information in der Aufgabe gegeben? An dieser Stelle sollte es eine Variable sein.’. Except for the highlighting in the task description, none of the other ways to help the user actually contribute to a solution but rather confuse a novice. More likely, the user will be driven away from his almost correct attempt to solve the task.
See B2Na: The incorrect use of an equation should be detected.

(d) obj(Onr,Typ,_,_,_), Typ = efh
Apart from the aforementioned semantical error using ‘Typ = efh’, the user has asked for the object IDs instead of street name and number.
With respect to the new error only there are 5 error messages. The user is faced with 7 errors which is pedagogically not desireable. The error messages themselves on their part do not contribute to a fast solution. It is proposed to use a constant in the first argument, where an anonymous variable is appropriate, and concerning the 3rd and 4th argument, the help/explanation proposes using a variable and a constant at the same time. The explanation askes whether the information about - in this case - the anonymous variables in the 3rd and 4th argument was given in the task.
See B2Nb: Using constants should not be proposed when anonymous variables are needed. Contradictions in help are to be avoided.

Unsatisfactory examples

none

4. Welches Haus (Strasse, Hausnummer) wurde mit einer Gewinnspanne von mehr als 100000 Euro weiterverkauft?
correct query: cannot be tested.
In order to receive semantically correct answers, the query has to ascertain that there is no transaction about a specific house between two transactions that fulfill the task. Otherwise the amount of money earned can be arbitrarily lower. Due to this nonmonotonic quality of the task and the inability of the system to process the goal ‘not’ I suspect the system of being unable to deal with queries that could possibly solve the task. It appears that a bug in the system prevents it from working in first place, even if the user’s query is simple like ‘obj(Obj,_,Str,Hnr,_), bew(_,Obj,_,Kaeufer,Preis1,_)’. Even if these errors in the system are corrected it seems to be quite inconveniant to solve a task like this with only a single line of Prolog code to be entered. For more difficult tasks a way to define help predicates and disjunctions seems to be necessary for a structured approach to the problem.

Positive examples

5. none

Negative examples

6. none

Unsatisfactory examples

(a) obj(Obj,_,Str,Hnr,_), bew(_,Obj,_,Kaeufer,Preis1,d(J1,M1,T1)), bew(_,Obj,Kaeufer,_,Preis2,d(J2,M
J2>J1, Preis1+100000 < Preis 2
The system does not give any information about the correctness of the solution. It copies the user's query and lists no error messages.

(b) obj(Obj,_,Str,Hnr,_), bew(_,Obj,_,Kaeufer,Preis1,_), bew(_,Obj,Kaeufer,_,Preis2,_),
Preis1+100000<Preis2
Exactly as in the previous example the system gives no information.

C Domain Bibliothek

1. Welche Bücher (Titel) muss der Leser mit der Lesernummer 2264 bis wann zurückgeben?
   correct query: ausleihe(2264,Sig,Rueck,_,_), buch(Sig,_,Titel,_)

   ### Positive examples

   (a) ausleihe(2264,Sig,Rueck,_,_)
   The user does not ask for the title but only for the signature.
   The missing subgoal is criticized in the error message. In the answer term, the words 'Welche Bücher (Titel)' are highlighted showing the user which detail of the task he has not complied with yet.

   (b) ausleihe(2264,Sig,Rueck,_,_), buch(Titel,_,Sig,_)
   The user has confused the argument positions of 'Titel' and 'Sig'.
   The error message 'Sig Diese Variable kann nicht innerhalb eines Ziels gleichzeitig mehrere Typen haben Siehe typischen Typkonfliktsfehler [Sig=titel, Sig=signatur]' does not designate the exact type of mistake but especially the help indicates that the user might have confused two types. This should allow the user to specifically search for the mistake he made.
   Except for the misleading advice to look up the type conflict, this help is satisfactory.

   (c) ausleihe(Lnr,S,Rueck,_,_), buch(S,_,Titel,_)
   Instead of using the constant 2264 the user applied the variable 'Lnr'.
   The answer term and the error message clearly propose reviewing the task and finding the correct constant.
   Mentioning that this query is slightly too general could perfectionate the diagnostics.

   (d) vorbestellung(2264,S,Rueck), buch(S,_,Titel,_)
   An incorrect first subgoal has been used.
   The answer term highlights the words 'bis wann zurückgeben' and the error messages state that another subgoal is needed and the subgoal 'vorbestellung' might be superfluous. This should allow the user to notice that he has to change the first subgoal and not just omit it.
   The particular help messages should be expressed more clearly. Otherwise the system performs well on this problem.

   ### Negative examples

   (a) ausleihe(2264,Sig,Rueck,_,_), buch(Sig,_,_,_)
   Somehow the user found the right subgoal but maybe forgot to ask for the title itself.
   The incorrect anonymous variable is highlighted but the error messages contradictorily propose applying a variable and a constant.
   The contradiction has to be resolved.

   (b) ausleihe(2264,_,Rueck,_,_), buch(_,_,Titel,_)
   The user did not link the two subgoals.

The two incorrect anonymous variables are highlighted and if the error message would not propose constants and variables simultaneously, the diagnostics would form a valueable help.
See above example.

(c) ausleihe(2264,S,_,_,_), buch(S,_,Titel,_)
The date of return was omitted by the user.
The answer term highlights the words ‘bis wann’ thus showing what the user forgot. Like in above examples, the error messages contradict each other.

Unsatifactory examples

(a) buch(Sig,_,Titel,_,X), ausleihe(2264,Sig,Dat,_,_,c664)
Another slot in ‘buch’ as well as an additional constant in ‘ausleihe’ have been invented by the user.
The system is unable to process the query and returns an interface in disarray.

(b) buch(Sig,_,Titel,_),ausleihe(2264,Sig,Rueck,_,_)
The subgoals have been alternated which causes the query to be rather inefficient but still correct.
The system neglects the inefficiency and declares the solution to be correct.
Pointing out the more efficient solution would be a nice feature. For a novice user this issue can be neglected, however.

(c) ausleihe(2264,S,d(J,M,T),_,_), buch(S,_,Titel,_)
Instead of a single variable for the date, the user decomposed it’s structure and asks for year, month and date specifically.
The diagnostics does not accept this solution as being correct. The error messsage questions whether the information was given in the task (which one could say is given there indeed) and proposes a variable. Maybe the user will be confused but following the advice should lead to the correct solution.
This solution is not the simplest one but it cannot be foreclosed that a beginner engages the task this way. Therefore the system should be able to recognize the correctness.

(d) ausleihe(4611,S,Rueck,_,_), buch(S,_,Titel,_)
A wrong constant was used.
The system does not give a feedback but rather shows the interface in disorder (see A3))

(e) ausleihe(3167,S,Rueck,_,_), buch(S,_,Titel,_)
A wrong constant which is used in other instances of ‘ausleihe’ was applied.
The system highlights the wrong constant but does not give an error message.
An advice to review the task in order to find the correct constant should be expected.

(f) ausleihe(2264,s,Rueck,_,_), buch(S,_,Titel,_)
The variable for signature has been written with a lower case ‘s’.

The system does not give feedback and only returns an interface in disorder.

2. Ist das Buch mit der Signatur c5674 vorbestellt?
correct query: vorbestellung(_,c5674,_)

Positive examples

(a) vorbestellung(_,Sig,_)
Instead of the constant a variable has been used.
The variable is highlighted and using a constant is proposed. The user is encouraged to review the task description which should easily enable him to correct the query.

Negative examples

(a) vorbestellung(_,c55674,_)
The constant was misspelled.
The answer term highlights the wrong constant but the error message proposes that if it should be a variable, it is supposed to be written with a capital letter. This confuses the novice user.
Error detection for constants is necessary.

Unsatisfactory examples

(a) vorbestellung(_,Sig,_), Sig = c5674
This is another (slightly too complicated) version of the correct query. The system does not give any information. Neither does it judge the query correct nor does it point at a mistake.
As novice users might possibly infer this solution from other examples they learnt, the system should be able to process this variant correctly.

3. Wo steht das Buch mit der Signatur a3325 normalerweise?
correct query: buch(a3325,_,_,Ort).

Positive examples

(a) a(X,Y)
The user - not knowing where to start - just entered anything which could be syntactically correct.
The system does not highlight the incorrect predicate ‘a’ but the error message states ‘Fehler kann nicht identifiziert werden. Deine Anfrage passt nicht zur Datenbank. Bitte prüfe die Prädikate nach.’. This is a general error message but still it helps the user to reconsider which predicate to use.
Improved highlighting would be positive. The statement that the error could not be identified is confusing and should be corrected.

(b) buch(X,Y)
As a next step towards the solution, the user found the right predicate name but uses wrong and too few arguments.
On first try, the system just returned the contorted interface with no further output. On second try, the variable 'Y' is highlighted while the error message states that the first subgoal has more slots to be filled. This should drive the user to review the description of the predicate 'buch'.
The system error on first try has to be corrected.

(c) buch(b3948,_,_,A)
The wrong constant was used while there are ground facts in the database in which this constant is used.
The wrong constant is detected and highlighted and the system proposes reviewing the task for the correct constant.
This diagnostics should also work for unknown incorrect constants.

(d) buch(Ort,_,_,a3325)
The argument positions were confused.
The wrong arguments are not highlighted but at least the error message reads 'a3325 Diese Konstante befindet sich an der falschen Argumentstelle. Bitte korrigiere es.' which definitely helps the user to find his mistake. The system works similar for wrong constants used in other ground facts.
See above.

Negative examples

(a) buch(X,Y,Z,A)
The sufficient number of terms is supplied as a step following the systems hints in C3Pb.
As X, Y and Z are incorrect, they are highlighted. For all of them the system uniformly proposes looking for and applying the appropriate constant. While for X there is one in the task description, the user might search for long time and in vain for information about Y and Z.
Hints concerning should-be constants and should-be anonymous variables should be different. Solving A1 probably fulfills this requirement.

Unsatisfactory examples

4. none

5. Welcher Leser (Name, Vorname) sind vor dem 1.1.1943 geboren?
correct query: leser(_,Name,Vorname,_,Jahr), Jahr < 1943

Positive examples

(a) leser(_,Name,Vorname,_,1943), Jahr < 1943
The constant 1943 was used instead of the appropriate variable.

In the answer term the words ‘geboren’ and ‘1943’ are highlighted. The error term states: ‘Jahr Diese Variable ist noch nicht instanziiert. Diese Variable sollte instanziiert sein, bevor sie für eine arithmetische Operation verwendet wird’ and ‘1943 War diese Information in der Aufgabe gegeben? An dieser Stelle sollte es eine Variable sein.’. Stating that ‘1943’ is incorrect and not given in the task can cause confusion but the hints concerning the variable ‘Jahr’ support the user in finding his error before dealing with that matter. The help saying that ‘Jahr’ is not instantiated yet is especially useful.
See B2Na: Constants given in the task but applied in a wrong subgoal or argument position should not be criticized as not given.

(b) leser(_,Name,Vorname,_,Jahr)
The user forgot to compare ‘Jahr’ with 1943.
The system highlights the date in the task description and states that another subgoal is required. This forms a good and apropriate help to the user.

(c) leser(_,Name,Vorname,_,Jahr) Jahr <1943
The user forgot to seperate the subgoals by a comma.
Surprisingly, the first (anonymous) variable is highlighted. Apart from that, the error message states that a comma is missing.
The highlighting should be corrected.

(d) leser(_,Name,Vorname,_,Jahr), Jahr <1944
The constant was misspelled.
The error term suggests having a close look at the inequation again which should easily enable the student to spot his mistake.

Negative examples

(a) leser(_,Name,Vorname,_,Jahr) ,Jahr <1943
Correcting the above mistake, the comma was placed wrong.
The system cannot process the query and states that there are too many syntax errors.
In this case of a little slip a more precise error description would be desireable.

Unsatisfactory examples

(a) leser(_,Name,Vorname,_,Jahr), Jahr =1943
The inequation was made an equation.
This drove the system into a probably infinite loop. There is no result even after minutes of calculation.
See B2Na: When inequations are required an equation should be considered to be relating to it. Thus using an inquation instead should be recommended by the system.

6. Welche Bücher (Titel) muss der Leser mit dem Nachnamen Heller bis wann zurückgeben?
correct query: buch(Sig,_,Titel,_), ausleihe(Lnr,Sig,Rueckgabe,_,_), leser(Lnr,hiller,_,_,_)

Positive examples

7. none

Negative examples

8. none

Unsatisfactory examples

(a) buch(_,_,Titel,_), ausleihe(_,_,Rueckgabe,_,_), leser(_,hiller,_,_,_)
The user omitted all links between the subgoals.
The slots in which variables linking the subgoals are needed are highlighted. Unfortunately, there are 8 error messages proposing using constants and variables at the same time. The user is adviced to review the task for filling these slots. This is not helpful and 8 messages are probably a number too large for the user to take into account.
A general cue indicating that the subgoals have to be linked in some way would be most desireable. This would also solve the problem of too many error messages at one time.