eXpander: A Graphical User Interface to Expander for the X Window System

Markus Heller, Peter Padawitz

{heller,padawitz}@cs.uni-dortmund.de
http://ls5.cs.uni-dortmund.de
University of Dortmund

September 11, 1997

Contents

Expander is a proof and test environment for algebraic data type specifications and functional-logic programs written in Standard ML and described in [Pad 94]. The description employs a NeXTSTEP interface, while the GUI eXpander presented here was designed for the X window system on the basis of Tcl/Tk.

The architecture of the environment underlying eXpander is presented in Section 1. Section 2 introduces the layout of eXpander and the various windows it consists of. The selection of items is explained in Section 3. In Section 4 we describe how to use the menus and windows of eXpander. The handling of error messages is discussed in Section 5.

1 Architecture

eXpander consists of an ML extension of Expander and a Tcl/Tk based program that provides the whole graphical functionality of the interface. The Tcl/Tk application is interpreted by a Tcl/Tk-Interpreter (called **wish**), which communicates with the X server, which manipulates graphics on the screen.

The Tcl/Tk application establishes a communication channel between ML and the GUI. For instance, pushing a button in eXpander's menus causes, via this channel, the Expander program to execute the corresponding command. Conversely, Expander uses the channel to make user output visible within the windows of eXpander.

The execution of the Expander program underlying the interface is hidden from the user. The ML Interpreter is run as a subprocess of the Tcl/Tk application.

2 Layout of eXpander

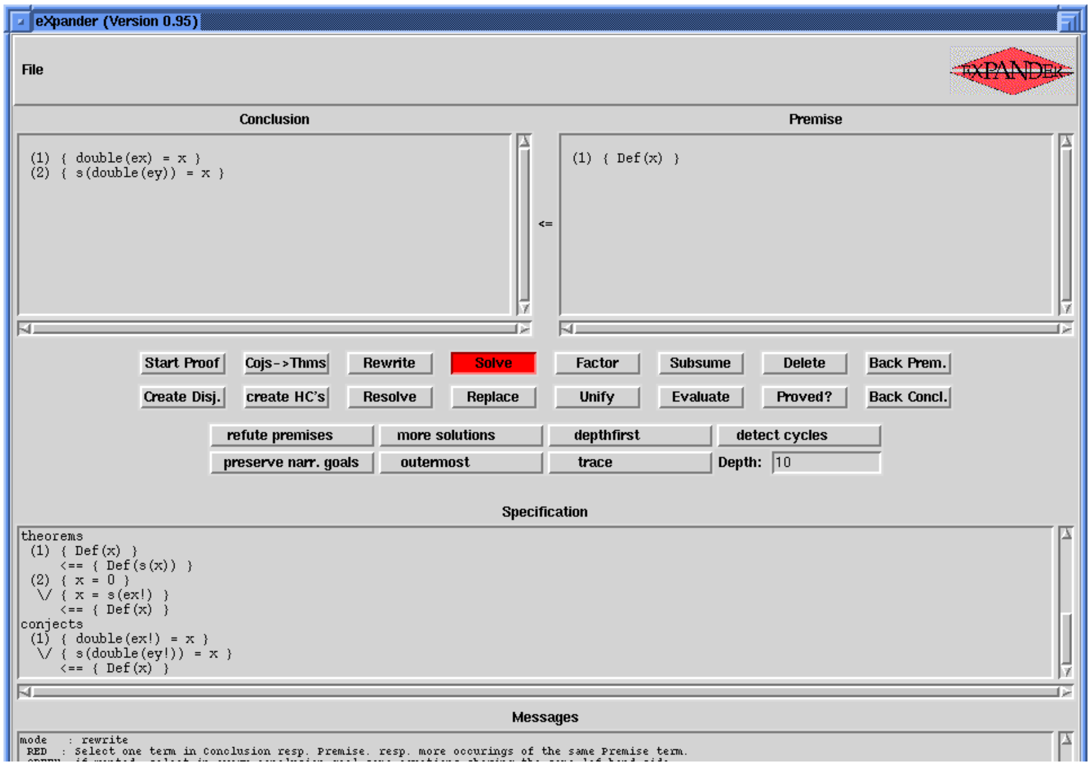

Figure 1. The eXpander environment

2.1 The menu *File*

The menu **File** is located at the upper left corner of the main window. Here the user finds all commands related to the handling of files. The user may load the specification of an abstract datatype or a program from disk or save several parts of the current proof to disk. Furthermore the eXpander session can be aborted by using the **Quit** button located in this menu.

2.2 The window *Specifications*

eXpander displays the currently loaded specification in the **Specifications** window located in the middle of the main window. Besides the axioms and theorems of the given specification the unproved conjectures are given here.

The proof of such an unproved conjecture can be performed stepwise using the commands provided by the Expander system.

Figure 2. The menu *File*

2.3 The windows *Conclusion* and *Premise*

While performing a proof the current sequent is displayed in these windows which can be found at the top of the main window. The current conclusion is displayed is the window **Conclusion** while the window **Premise** shows the premise of the current sequent.

2.4 Command buttons

Located in the center of the main window between the **Specifications** window and the windows **Conclusion** and **Premise** are the command buttons. Every command button performs a single command of the Expander system. with these commands the user may manipulate some previously selected terms, atoms or goals of the current sequent. Several commands provide additional options which will be displayed right below the command buttons, if needed.

2.5 The window *Messages*

eXpander displays all messages to the user in the **Messages** window located at th ebottom of the main window. The complete Expander session is logged here including all executed commands and their results as well as Expander's error messages or help about the commands.

3 Selecting terms, atoms, goals and clauses

Many commands require the selection of some terms, atoms, goals or clauses. Items may be marked in three ways that are distinguished by different cursor symbols:

- arrow (red colour): most commands expect their operands selected with the arrow.

Additionally, the commands **Resolve**, **Replace** and **Rewrite** take two other types of marking:

- spraycan (green colour): marking single elements that are used for inserting or rewriting.
- double arrow (blue colour): selecting pairs of elements indicating a correspondence between them.

3.1 Changing the marking type and the cursor symbol

By clicking the right mouse-button the user may change between all the types of marking which are allowed in the current situation. For example, if the user is allowed to change between arrow and spraycan marking, repeatedly clicking the right mouse-button alternates between the two cursor symbols.

3.2 Selecting a single item

The user may select a single item in red or green colour only. By clicking the right mouse-button the user first chooses the arrow resp. spraycan symbol.

Then one may select a single item by simply moving the cursor close to the beginning of a term, atom, goal or clause and pressing the left mouse-button. Releasing the button adjusts the end of the selection and causes eXpander to highlight the smallest type of formula (term, atom, goal or clause) enclosing the cursor position.

One may alternatively select a formula by highlighting the entire selection.

If a single selection overlaps another previously made single selection, eXpander removes the older one and only keeps the newer selection assuming that the user now corrects his previous selection.

Note that a single selection may overlap a pair selection.

3.3 Pairing two single items

When performing the command **Rewrite** the user has to pair two items indicating a correspondece between them. By clicking the right mouse-button the user first chooses the doublearrow symbol for pairing-mode.

Two single items selected immediately one after another pairs them together. One may imagine this as putting them in a bag. each time two items are in the bag they are removed and formed into a pair. Both items are highlighted in blue colour and get prefixed with the same upper-case letter to indicate their correspondence.

Deleting one member of a pair destroys their binding and puts back the remaining item into the bag. If then again two items are in the bag they form a new pair.

In general one item is a term or atom of the current proof (displayed in the windows **Premise** and **Conclusion**, the other one is an atom of a theorem or conjecture and they are either both in the conclusion or both in the premise.

If more than one pairing is made, the following conditions must be ensured:

- All items in theorems resp. conjectures denote different atoms of the same goal of a clause and all items selected in the current proof denote different premise terms or atoms (resp. conclusion terms or atoms) of a single goal.
- All items in theorems resp. conjectures denote atoms in different goals of a clause while all items in the current proof denote terms or atoms distributed over several conclusion goals.

3.4 Deleting a selection

Every selection may be deleted by selecting it again (un-marking it) or even simpler by just clicking the left mouse-button over it without highlighting anything (in other words the zero letter selection replaces the older, previously made selection of the item as a correction).

4 Commands

In the sequel we describe how to use the commands of Expander within the interface.

4.1 The menu *File*

Click the menubutton **File** in the upper left corner of the main window.

4.1.1 The button *parse Spec*

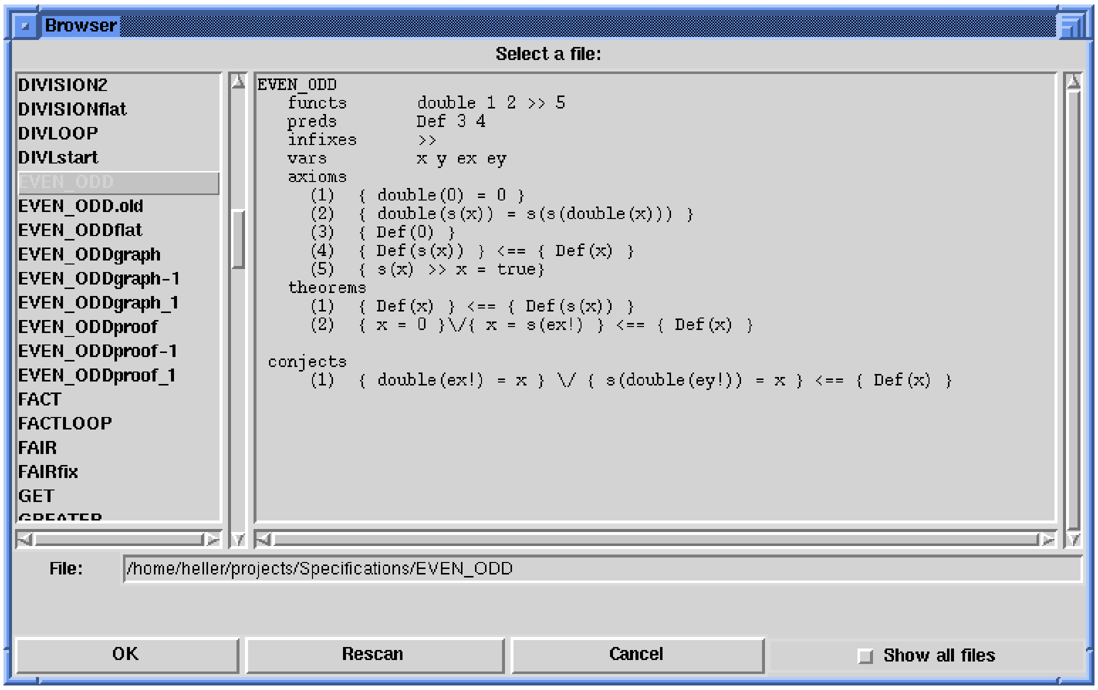

Figure 3. The browser

This command can be found in the **File** menu.

A file browser appears on the screen, the contents of the various files in the current directory can be previewed (cf. Fig. 3).

The user may leave the browser by selecting **Cancel**, rescan the current directory by clicking **Rescan** or choose the displayed file for parsing by clicking **Ok**. (Additonally he may click **show all files** to list all of the files in the directory inclusing .dot-files.)

If the file contains a valid Expander specification it will be parsed according to the grammar described in the Expander's user manual (cf. [Pad 94]). Furthermore the specification is assigned to the specification component of Expander's system state and also entered into the file **PARSE** in the current working directory. After parsing the file browser is removed from the screen.

If the specification is incorrect then the part of it that could be successfully parsed is displayed in the **Messages** window together with error messages. The file **PARSE** also contains the successfully parsed part of the specification.

4.1.2 The button *store flat*

This command can be found in the **File** menu. The flat version of the current specification is entered into the file SPflat in the Specifications directory.

4.1.3 The button *store Graph*

This command can be found in the **File** menu. The file SPn in the Graphs directory is generated and each solved goal in the current proof is entered into it.

4.1.4 The button *store Proof*

This command can be found in the **File** menu. The file SPn in the Graphs directory is generated and the current proof is entered into it."

4.1.5 The button *store Tree*

This command can be found in the **File** menu. Select a goal in the current conclusion or premise using red colour. The tree representation of this goal is entered into the file CONCn.eps resp. PREMn.eps.

4.1.6 The button *Preferences*

This command can be found in the **File** menu. Expander stores the specifications, the generated proof files as well as the generated graphs in three different directories to be created by the user.

4.1.7 The button *Quit*

This command can be found in the **File** menu. The user may confirm the termination by moving the cursor over the **yes** field appearing on the screen just right to the Quit button. Doing so one quits the complete eXpander session. If one does not want to quit, the **no** field must be clicked.

4.2 Commands for starting and modifying a proof

4.2.1 The button *Start Proof*

Click **Start Proof**. An additional option **Guard** is displayed below the command buttons.

Figure 4. The additional option for the *Start Proof* command

To start a proof one may alternatively prove (1) a single Gentzen clause or (2) several Horn clauses simultaneously.

(1) Select a single conjecture in the **Specifications** window.
Let Guard=1.

Click again **Start proof**.

The **Conclusion** window is initialized with the conclusion goals of the selected conjecture and the premise goals of it are taken into the **Premise** window.

(2) Select several conjectures, which are guarded Horn clauses and thus have a single conclusion goal and whose first, say n, premise atoms are identical.
Let Guard=n.
Click again**Start Proof**.

The **Conclusion** window is initialized with the goal consisting of all identical atoms. The **Conclusiuon** window takes as many goals as conjectures were selected, each of them containing the remaining atoms of premise and conclusion together of that conjecture, i.e. the non-identical premise atoms of that conjecture plus its conclusion atoms.

4.2.2 The button *Resolve*

Click **Resolve**. Additional options **preserve redex** and **refute premises** are displayed below the command

Figure 5. The additional options for the *Resolve* command

buttons.

Alternatively one may (1) expand a **conclusion or premise atom** upon all applicable axioms, (2) expand **conclusion atoms** of a single goal upon a theorem or conjecture, (3) expand **conclusion atoms** distributed over several goals upon a theorem or conjecture, (4) contract **premise atoms** of a single goal upon a theorem or conjecture.

(1) Select a single atom $P(t)$ in the current conclusion in red color.

If **Refute Premises** is clicked, then only those axioms for P are applied to $P(t)$, which have a *feasible* premise g, i.e. the instance of g generated by the *Resolve* cannot be refuted within 5 expansion steps. Hence **Refute Premises** may restrict the number of applicable axioms.

Click again **Resolve**.

(2) Choose a conjecture or theorem that has only a single conclusion goal. Using blue colour, pair atoms of the goal with atoms in a single goal of the current conclusion. Click again **Resolve**.

(3) Choose a conjecture or theorem $gs \Leftarrow h$. Using blue colour, pair a single atom in each goal of gs with an atom in disjoint goals of the current conclusion. Click again **Resolve**.

(4) Choose a conjecture or theorem $gs \Leftarrow h$. Using blue colour, pair each atom of h with an atom in a single goal of the current premise. Click again **Resolve**.

Clicking **Preserve Redex** saves the current conclusion resp. premise.

4.2.3 The button *Rewrite*

Click **Rewrite**. Again, additional options **preserve redex** and **refute premises** are displayed below the command buttons.

Alternatively, one may (1) expand a **conclusion or premise term** upon all applicable axioms, (2) expand a **conclusion term** upon a theorem or conjecture, (3) expand **conclusion terms** distributed over several goals upon a theorem or conjecture, (4) contract a **premise term and premise atoms** of a single goal upon a theorem or conjecture.

(1) Using red colour, select a single term in the current proof. Click (optionally) **Refute Premises** (see above). Click again **Rewrite**.

(2) Choose a conjecture or theorem that has a single conclusion goal. Using blue colour, pair an equation of the goal with a term in the current conclusion. Click again **Rewrite**.

(3) Choose a conjecture or theorem $gs \Leftarrow h$. Using blue colour, pair an equation in each goal of gs with a term in a single goal of the current conclusion. All equations are supposed to have the same right-hand side. Click again **Rewrite**.

(4) Choose a conjecture or theorem $gs \Leftarrow h$. In blue colour, pair each atom of h with an atom in a single goal of the current premise. In green colour, select in each goal of gs, say, n equations with the same left-hand side. In red colour, select n occurrences of the same term in a single goal of the current conclusion. Click **DO**.

Clicking **Preserve Redex** saves the current conclusion resp. premise.

4.2.4 The button *Solve*

Click **Solve**. Additonal options **refute premises**, **preserve narrowed goals**, **more solutions**, **outermost**, **depthfirst**, **trace**, **detect cycles** and **Depth** are displayed below the command buttons.

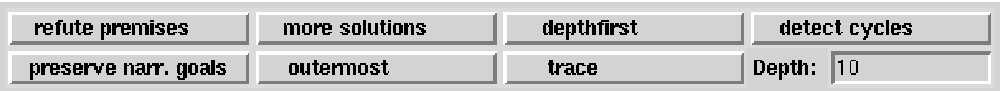

Figure 6. The additional options for the *Solve* command

A single goal may be solved in at most n steps. For this purpose the goal is transformed by a sequence of greedy expansion steps upon axioms in order to derive solved goals from the goal.

Select a goal in the current proof. Choose some of the option parameters if wanted. Especially enter the **Depth** n of the expansion (cf. Fig. 6). Click again **Solve**. If possible, the selected goal will be solved in at most n steps, .

4.3 Commands for simplification

4.3.1 The button *Factor*

In red colour, select two atoms in a single goal of the current conclusion. Click again **Factor**. Both atoms will be factored if possible.

4.3.2 The button *Unify*

In red colour, select an equation of the current conclusion. Click again **Unify**. The sides of the selected equation will be unified if possible.

4.3.3 The button *Replace*

In red colour, select up to four terms in a single goal of the current premise or conclusion that should be replaced. In green colour, select an equation $t \equiv t'$ in the same goal such that t resp. t' agrees with the terms to be replaced. Click again **Replace**. The selected terms will be replaced by t' resp. t.

4.3.4 The button *Delete*

In red colour, select several atoms and goals in the current proof. Click again **Delete**. The selected atoms and goals will be removed from the current proof.

4.3.5 The button *Subsume*

The current goal set is replaced by a subset of goals that is subsumed by the goal set.

4.3.6 The button *Evaluate*

In red colour, select several goals all of which belong either to the current conclusion or to the current premise. Click again **Evaluate**. The selected goals are simplified by Expander's simplifier.

4.4 Commands for stopping a proof

4.4.1 The button *Proved*

Checking whether the current proof is complete.

If the query is successful, then Expander answers (in the **Messages** window) with

"conjectures ... have been proved.".

Otherwise the message is

"conjectures ... have not yet been proved.".

4.4.2 The button *Create Disj.*

A list of Gentzen clauses is constructed from the current conclusion and premise and added to the list of conjectures of SP (cf. [Pad 94], 4.6(3a)).

4.4.3 The button *Create HCs*

A list of Horn clauses is constructed from the current conclusion and premise and added to the list of conjectures of SP (cf. [Pad 94], 4.6(3b)). This works only if the current premise consists of a single goal. The conclusion atoms that shall be shifted to the premises of the Horn clauses are to be selected in red colour. Click again **Create HCs**.

4.4.4 The button *Cojs to Thms*

The actual conjectures are moved to the list of theorems of SP.

4.5 Commands for backtracking

Pushing the button *Back Concl.* resp. *Back Prem.* causes eXpander to backtrack the proof to the immediate predecessor of the current conclusion resp. premise.

References

[Pad 94] P. Padawitz, *Expander: A System for Testing and Verifying Functional-Logic Programs*, Report, FB Informatik, University of Dortmund 1994-1996, http://ls5.cs.uni-dortmund.de/~peter/ExpaTex.ps.gz