Hints for Using JAPE for Natural Deduction Proofs

Robert Keller

13 September 2007

Load the *theory* first, then reload any proofs that you may have saved in that theory.

There are two forms of **selection**:

- Formula selection, by clicking. Shows as an outlining box.
- Sub-formula selection, by option/alt-dragging. Shows as a highlight.

Sub-formula selection is used for unification (under the edit menu). This can sometimes be avoided by using the hyp rule.

"**Available**" means proved (from above).
"**Goal**" means to be proved.
"**NA**" means not applicable.

Rule	**Backward**	**Forward**	**Example ,** either after rule or Before After
∧ Intro	Use when goal is a conjunction.	Use to conjoin to available formulas.	5: S(i) ∀ elim 2,4 6: R(i) ∀ elim 3,4 7: R(i)∧S(i) ∧ intro 6,5
∧ Elim	NA	Use to separate a conjunction.	7: R(i)∧S(i) ∀ elim 1,6 8: S(i) ∧ elim 7
∨ Intro	*Sometimes* used when goal is a disjunction. Loses information, however.	Use to derive disjunctive goal from an available formula. Invents the other half of the goal, which must be unified in a separate step.	3: R(i) assumption 4: ∃y.R(y) ∃ intro 3,2.1 5: ∃y.R(y)∨∃z.S(z) ∨ intro 4
∨ Elim	NA	Use to derive a specific goal from a disjunction. Creates two sub-proofs for the goal.	2: actual i, R(i)∨S(i) assumptions 3: R(i) assumption 4: ∃y.R(y) ∃ intro 3,2.1 5: ∃y.R(y)∨∃z.S(z) ∨ intro 4 6: S(i) assumption 7: ∃z.S(z) ∃ intro 6,2.1 8: ∃y.R(y)∨∃z.S(z) ∨ intro 7 9: ∃y.R(y)∨∃z.S(z) ∨ elim 2.2,3-5,6-8

→ Intro	Use when goal is an implication. Creates sub-proof.	NA	2: ∀y.R(y) assumption 3: actual i assumption 4: R(i)→S(i) ∀ elim 1,3 5: R(i) ∀ elim 2,3 6: S(i) → elim 4,5 7: ∀z.S(z) ∀ intro 3-6 8: ∀y.R(y)→∀z.S(z) → intro 2-7
→ Elim	NA	Use when an implication and the formula on the right are both available.	4: R(i)→S(i) ∀ elim 1,3 5: R(i) ∀ elim 2,3 6: S(i) → elim 4,5
¬ Intro	Use when goal is a negation. Creates sub-proof (with conclusion ⊥).	NA	1: ∀x.R(x) . . . 2: ¬∃y.¬R(y) ——— 1: ∀x.R(x) premise 2: ∃y.¬R(y) assumption . . . 3: ⊥ 4: ¬∃y.¬R(y) ¬ intro 2-3
¬ Elim	From goal ⊥, creates a pair of goals φ and ¬φ, which must be unified in a separate step.	Use when a pair of goals φ and ¬φ are available.	**Backward:** 1: ¬∃x.R(x) premise 2: actual i assumption 3: R(i) assumption . . . 4: ⊥ 5: ¬R(i) ¬ intro 3-4 6: ∀y.¬R(y) ∀ intro 2-5 ——— 1: ¬∃x.R(x) premise 2: actual i assumption 3: R(i) assumption . . . 4: _B1 . . . 5: ¬_B1 6: ⊥ ¬ elim 4,5 7: ¬R(i) ¬ intro 3-6 8: ∀y.¬R(y) ∀ intro 2-7 Now unify ¬_B1 with the top goal, using hyp. ——— **Forward:** 1: ∀x.R(x) premise 2: ∃y.¬R(y) assumption 3: actual i, ¬R(i) assumptions 4: R(i) ∀ elim 1,3.1 . . . 5: ⊥

			1: ∀x.R(x) premise 2: ∃y.¬R(y) assumption 3: actual i, ¬R(i) assumptions 4: R(i) ∀ elim 1,3.1 5: ⊥ ¬ elim 4,3.2
Contra (classical) same as RAA	With goal φ, creates a sub-proof, with ¬φ as assumption and ⊥ as goal.	NA	4: ¬R(i) assumption 5: ∃y.¬R(y) ∃ intro 4,3 6: ⊥ ¬ elim 5,2 7: R(i) contra (classical) 4-6
Contra (constructive) (same as ⊥ Elim)	Introduces goal ⊥, from which the current goal is derived. Often followed by ¬ Elim backward, which creates formulas to be unified.	Introduce goal from ⊥, which must be unified as a separate step.	1: ¬E premise 2: E assumption 3: ⊥ ¬ elim 2,1 4: F contra (constructive) 3 5: E→F → intro 2-4
hyp	Use to unify a goal with an available formula.	Use to unify an available formula with a goal.	1: ¬∃x.R(x) premise 2: actual i assumption 3: R(i) assumption . . . 4: _B1 . . . 5: ¬_B1 6: ⊥ ¬ elim 4,5 7: ¬R(i) ¬ intro 3-6 8: ∀y.¬R(y) ∀ intro 2-7 1: ¬∃x.R(x) premise 2: actual i assumption 3: R(i) assumption . . . 4: ∃x.R(x) 5: ⊥ ¬ elim 4,1 6: ¬R(i) ¬ intro 3-5 7: ∀y.¬R(y) ∀ intro 2-6
truth	Introduce ⊤ as a goal.	NA	1: ⊤ truth
∀ Intro	From a ∀ goal, open sub-proof with a new variable representing an arbitrary element.	NA	1: ¬∃x.R(x) . . . 2: ∀y.¬R(y) 1: ¬∃x.R(x) premise 2: actual i assumption . . . 3: ¬R(i) 4: ∀y.¬R(y) ∀ intro 2-3

∀ Elim	NA	Derive a goal φ[i/x] from an available formula ∀xφ by selecting i. Note: The ∀ formula and the variable must both be selected.	1: ∀x.R(x) premise 2: ∃y.¬R(y) assumption 3: actual i, ¬R(i) assumptions . . . 4: ⊥ 5: ⊥ ∃ elim 2,3-4 6: ¬∃y.¬R(y) ¬ intro 2-5 1: ∀x.R(x) premise 2: ∃y.¬R(y) assumption 3: actual i, ¬R(i) assumptions 4: R(i) ∀ elim 1,3.1 . . . 5: ⊥ 6: ⊥ ∃ elim 2,3-5 7: ¬∃y.¬R(y) ¬ intro 2-6
∃ Intro	From a goal ∃xφ and a selected variable i, introduce a sub-proof with a goal φ[i/x]. Note: The ∃ formula and the variable must both be selected.	NA	1: ¬∃x.R(x) premise 2: actual i assumption 3: R(i) assumption . . . 4: ∃x.R(x) 5: ⊥ ¬ elim 4,1 6: ¬R(i) ¬ intro 3-5 7: ∀y.¬R(y) ∀ intro 2-6 1: ¬∃x.R(x) premise 2: actual i assumption 3: R(i) assumption 4: ∃x.R(x) ∃ intro 3,2 5: ⊥ ¬ elim 4,1 6: ¬R(i) ¬ intro 3-5 7: ∀y.¬R(y) ∀ intro 2-6
∃ Elim	NA	From a ∃ goal, derive a goal with the variable instantiated to a new variable.	1: ∀x.R(x) premise 2: ∃y.¬R(y) assumption . . . 3: ⊥ 4: ¬∃y.¬R(y) ¬ intro 2-3 1: ∀x.R(x) premise 2: ∃y.¬R(y) assumption 3: actual i, ¬R(i) assumptions . . . 4: ⊥ 5: ⊥ ∃ elim 2,3-4 6: ¬∃y.¬R(y) ¬ intro 2-5

Using other theorems as lemmas:

- Select the place in the current proof that you want to apply the lemma.
- Select the lemma from the menu of conjectures. Note that the lemma has to be proved before it can be used.
- Click **Apply** in the conjectures menu.

Example:

1:	actual j, actual k, ∃x.¬R(x)	premises
2:	actual i, ¬R(i)	assumptions
3:	R(i)→R(j)∧R(k)	Theorem ¬E ⊢ E→F 2.2
4:	∃y.(R(y)→R(j)∧R(k))	∃ intro 3,2.1
5:	∃y.(R(y)→R(j)∧R(k))	∃ elim 1.3,2-4

You can use this approach, for example, to introduce LEM by selecting the three dots.

No non-empty universe assumption:

Unlike most texts, JAPE does not assume a non-empty universe. Thus you will not be able to prove some sequents unless you indicate that there is at least one element in the universe, by including actual j (where j is a variable representing an individual) as a premise.

Example:

1:	actual j, ∀x.R(x)	premises
2:	R(j)	∀ elim 1.2,1.1
3:	∃y.R(y)	∃ intro 2,1.1

Adding provisos:

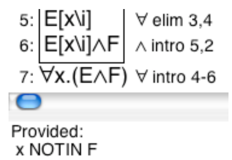
Sometimes we encounter sequents that require certain variables not be free in certain sub-formulas. An example is:

|– (∀x E) ∧ F ⇔ ∀x (E ∧ F)

which is provable iff x is not free in F. One way to have this proviso added for us is to unify a formula having a substitution with a formula not having it. Here is an example:

6: E[x\i]∧F
. . .
7: E[x\i]∧F[x\i]

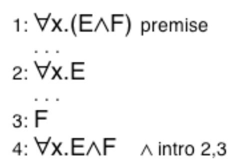
Lines 6 and 7 above would be the same if F had no free occurrences of x. If we unify them using the *hyp* rule, that proviso will be added:



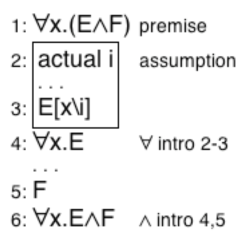
Notice that the provisos appear in a sub-window, but are not shown in the conjectures menu.

Upper vs. Lower Clicking

In some cases, a formula could be used either as a goal or as available. In this case, the choice is determined by clicking on the upper part of the formula vs. the lower part. In this proof, for example, ∀x.E can be used either way. To get the ∀I rule to be applied, we need to click on the upper part of ∀x.E.



We can then use ∀I:



Further information

There is a book of logic focused on JAPE:

Richard Bornat, *Proof and Disproof in Formal Logic – An Introduction for Programmers*, Oxford Texts in Logic, 2, Oxford University Press, 2005.