CS486 Problem Set 11 DUE: 5/01/03

1. We show that all functions that are representable in the theory $\mathcal{Q}$ are computable.

   Recall that an n-ary function $f : \mathbb{N}^n \to \mathbb{N}$ is representable in the theory $\mathcal{Q}$ if there is an $(n+1)$-ary predicate R_f, such that for all $x_1, \ldots, x_n \in \mathbb{N}$:

   - $f(x_1, \ldots, x_n) = y$ implies $\models_{\mathcal{Q}} \; R_f\left(\underline{x_1}, \ldots, \underline{x_n}, \underline{y}\right)$.
   - $f(x_1, \ldots, x_n) \neq y$ implies $\models_{\mathcal{Q}} \; \sim R_f\left(\underline{x_1}, \ldots, \underline{x_n}, \underline{y}\right)$.

   Furthermore, recall that the theory $\mathcal{Q}$ is finitely axiomatizable; hence, it is possible (i.e., computable) to enumerate all of the derivations that follow from the axioms of $\mathcal{Q}$. Therefore, to calculate $f(x_1, \ldots, x_n)$, it suffices to search for a y such that there exists a proof of $R_f(\underline{x_1}, \ldots, \underline{x_n}, \underline{y})$.

2. Let $B_1(x)$ and $B_2(x)$ be formulas in the language $\mathcal{Q}$ with x as the sole free variable. We show how to construct formulas G_1 and G_2 such that

$$\models_{\mathcal{Q}} G_1 \Leftrightarrow B_1(\ulcorner G_2 \urcorner) \quad \text{and} \quad \models_{\mathcal{Q}} G_2 \Leftrightarrow B_2(\ulcorner G_1 \urcorner)$$

Recall that $\mathcal{Q}$ is a theory in which *diag* is representable. Hence, by the Diagonal Lemma, for any formula $B(x)$ with x as the sole free variable, there is a formula G such that $\models_{\mathcal{Q}} G \Leftrightarrow B(\ulcorner G \urcorner)$.

Note that $B_1(\ulcorner B_2(x) \urcorner)$ is a formula with x as the sole free variable. Hence, there is a formula G such that $\models_{\mathcal{Q}} G \Leftrightarrow B_1(\ulcorner B_2(\ulcorner G \urcorner) \urcorner)$. Define $G_1 = G$ and $G_2 = B_2(\ulcorner G \urcorner)$. Then

$$\models_{\mathcal{Q}} G_1 \Leftrightarrow B_1(\ulcorner G_2 \urcorner) \quad \equiv \quad \models_{\mathcal{Q}} G \Leftrightarrow B_1(\ulcorner B_2(\ulcorner G \urcorner) \urcorner)$$

and

$$\models_{\mathcal{Q}} G_2 \Leftrightarrow B_2(\ulcorner G_1 \urcorner) \quad \equiv \quad \models_{\mathcal{Q}} B_2(\ulcorner G \urcorner) \Leftrightarrow B_2(\ulcorner G \urcorner)$$

3. Let $Prov$ be a provability predicate for the theory $\mathcal{Q}$ and X and Y be formulas in the language of $\mathcal{Q}$. Assume $\models_{\mathcal{Q}} Prov(\ulcorner X \urcorner) \supset Y$ and $\models_{\mathcal{Q}} Prov(\ulcorner Y \urcorner) \supset X$. We show that both X and Y are theorems in $\mathcal{Q}$.

   Recall the properties of a provability predicate:

   (P1) for all X, if $\models_{\mathcal{Q}} X$, then $\models_{\mathcal{Q}} Prov(\ulcorner X \urcorner)$

   (P2) for all X and Y, $\models_{\mathcal{Q}} Prov(\ulcorner X \supset Y \urcorner) \supset (Prov(\ulcorner X \urcorner) \supset Prov(\ulcorner Y \urcorner))$

   (P3) for all X, $\models_{\mathcal{Q}} Prov(\ulcorner X \urcorner) \supset Prov(\ulcorner Prov(\ulcorner X \urcorner) \urcorner)$

   (NOTE: (P1) is *not* equivalent to "for all X, $\models_{\mathcal{Q}} X \supset Prov(\ulcorner X \urcorner)$".)

   Recall that $\mathcal{Q}$ is a theory that can represent the computable functions. Hence, by Löb's Theorem, we have:

   (L) for all X, if $\models_{\mathcal{Q}} Prov(\ulcorner X \urcorner) \supset X$, then $\models_{\mathcal{Q}} X$

   Finally, recall that $\mathcal{Q}$ is a consistent theory; hence, the following *modus ponens* style theorems are true:

   (MP1) for all X and Y, if $\models_{\mathcal{Q}} X \supset Y$ and $\models_{\mathcal{Q}} X$, then $\models_{\mathcal{Q}} Y$

   (MP2) for all X and Y and Z, if $\models_{\mathcal{Q}} X \supset Y$ and $\models_{\mathcal{Q}} Y \supset Z$, then $\models_{\mathcal{Q}} X \supset Z$

   We reason as follows:

   (A1) $\models_{\mathcal{Q}} Prov(\ulcorner X \urcorner) \supset Y$, by assumption.

   (A2) $\models_{\mathcal{Q}} Prov(\ulcorner Y \urcorner) \supset X$, by assumption.

   (B1) $\models_{\mathcal{Q}} Prov(\ulcorner Prov(\ulcorner X \urcorner) \supset Y \urcorner)$, by (A1) and (P1).

   (B2) $\models_{\mathcal{Q}} Prov(\ulcorner Prov(\ulcorner Y \urcorner) \supset X \urcorner)$, by (A2) and (P1).

   (C1) $\models_{\mathcal{Q}} Prov(\ulcorner Prov(\ulcorner X \urcorner) \urcorner) \supset Prov(\ulcorner Y \urcorner)$, by (P2), (B1), and (MP1).

   (C2) $\models_{\mathcal{Q}} Prov(\ulcorner Prov(\ulcorner Y \urcorner) \urcorner) \supset Prov(\ulcorner X \urcorner)$, by (P2), (B2), and (MP1).

   (D1) $\models_{\mathcal{Q}} Prov(\ulcorner Prov(\ulcorner X \urcorner) \urcorner) \supset X$, by (C1), (A2), and (MP2).

   (D2) $\models_{\mathcal{Q}} Prov(\ulcorner Prov(\ulcorner Y \urcorner) \urcorner) \supset Y$, by (C2), (A1), and (MP2).

   (E1) $\models_{\mathcal{Q}} Prov(\ulcorner X \urcorner) \supset X$, by (P3), (D1), and (MP2).

   (E2) $\models_{\mathcal{Q}} Prov(\ulcorner Y \urcorner) \supset Y$, by (P3), (D1), and (MP2).

   (F1) $\models_{\mathcal{Q}} X$, by (E1) and (L).

   (F2) $\models_{\mathcal{Q}} Y$, by (E2) and (L).

   Thus, both X and Y are theorems in $\mathcal{Q}$.

4. A pair of natural numbers $\langle a, b\rangle$ is called a *stamps pair* if for all $n \geq a + b$ there are natural numbers i and j such that $n = i * a + j * b$.

   (a) We prove in λ-PRL that $\langle 3, 5\rangle$ is a stamps pair.

```
⊢ ∀n:int. 8 ≤ n ⇒ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (n = i * 3 + j * 5)
|                                                          by induction 1 8 10 3
|\
| 1. n:int
| 2. n < 8
| 3. 8 ≤ n+1 ⇒ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (n+1 = i * 3 + j * 5)
| ⊢ 8 ≤ n ⇒ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (n = i * 3 + j * 5)
| |                                                        by impR
| ...
| 4. 8 ≤ n
| ⊢ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (n = i * 3 + j * 5)
|                                                          by arith √
|\
| ⊢ 8 ≤ 8 ⇒ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (8 = i * 3 + j * 5)
| |                                                        by impR
| 1. 8 ≤ 8
| ⊢ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (8 = i * 3 + j * 5)
| |                                                        by exR 1 1
| ...
| ⊢ (0 ≤ 1) ∧ (0 ≤ 1) ∧ (8 = 1 * 3 + 1 * 5)
|                                                          by arith √
|\
| ⊢ 8 ≤ 9 ⇒ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (9 = i * 3 + j * 5)
| |                                                        by impR
| 1. 8 ≤ 9
| ⊢ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (9 = i * 3 + j * 5)
| |                                                        by exR 3 0
| ...
| ⊢ (0 ≤ 3) ∧ (0 ≤ 0) ∧ (9 = 3 * 3 + 0 * 5)
|                                                          by arith √
|\
| ⊢ 8 ≤ 10 ⇒ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (10 = i * 3 + j * 5)
| |                                                        by impR
| 1. 8 ≤ 10
| ⊢ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (10 = i * 3 + j * 5)
| |                                                        by exR 0 2
| ...
| ⊢ (0 ≤ 0) ∧ (0 ≤ 2) ∧ (10 = 0 * 3 + 2 * 5)
|                                                          by arith √
 \
  1. n:int
  2. 10 < n
  3. 8 ≤ n-3 ⇒ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (n-3 = i * 3 + j * 5)
  ⊢ 8 ≤ n ⇒ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (n = i * 3 + j * 5)
  |                                                        by impL 3
  |\
  | ...
  | ⊢ 8 ≤ n-3
  |                                                        by arith √
  |
   \
    ...
    3. 8 ≤ n-3
    4. ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (n-3 = i * 3 + j * 5)
    ⊢ 8 ≤ n ⇒ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (n = i * 3 + j * 5)
    |                                                      by exR 4
```

```
|
...
4. i:int
5. j:int
6. (0 ≤ i) ∧ (0 ≤ j) ∧ (n-3 = i * 3 + j * 5)
⊢ 8 ≤ n ⇒ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (n = i * 3 + j * 5)
|                                                          by andL 6
...
6. 0 ≤ i
7. 0 ≤ j
8. n-3 = i * 3 + j * 5
⊢ 8 ≤ n ⇒ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (n = i * 3 + j * 5)
|                                                          by impR
...
9. 8 ≤ n
⊢ ∃i,j:int. (0 ≤ i) ∧ (0 ≤ j) ∧ (n = i * 3 + j * 5)
|                                                          by exR i+1,j
...
⊢ (0 ≤ i+1) ∧ (0 ≤ j) ∧ (n = (i+1) * 3 + j * 5)
                                                           by arith √
```

(b) We describe the algorithm implicitly contained in the proof of (a) in λ-PRL notation.

```
stampsi(n:int):int ≡  n => 0
                      8 => 1
                      9 => 3
                      10 => 0
                      n => stampsi(n-3) + 1

stampsj(n:int):int ≡  n => 0
                      8 => 1
                      9 => 0
                      10 => 2
                      n => stampsj(n-3)
```

(c) If $\langle a, b\rangle$ is a stamps pair (such that $a \leq b$), then

- $a = 1$, or
- $a = 2$ and $b = 1 \bmod 2$, or
- $a = 3$ and $b = 4$, or
- $a = 3$ and $b = 5$

5. We prove in λ-PRL that every non-empty list of integers has a maximal element.

   $\forall$l:list. ($\sim$(l=[]) $\Rightarrow$ $\exists$i:int. (i$\in$l $\wedge$ $\forall$j:int. (j$\in$l $\Rightarrow$ j$\leq$i)))

   We give a λ-PRL definition of the predicate i$\in$l:

   i$\in$l $\equiv$ $\exists$k:int. (index(k,l)=i $\wedge$ 0$\leq$k $\wedge$ k<length(l))

   We prove the above statement λ-PRL's refinement logic:

```
⊢ ∀l:list. (∼(l=[]) ⇒ ∃i:int. (i∈l ∧ ∀j:int. (j∈l ⇒ j≤i)))
|                                                          by induction x::l
|\
| ⊢ ∼([]=[]) ⇒ ∃i:int. (i∈[] ∧ ∀j:int. (j∈[] ⇒ j≤i))
| |                                                        by impR
| 1. ∼([]=[])
| ⊢ ∃i:int. (i∈[] ∧ ∀j:int. (j∈[] ⇒ j≤i))
| |                                                        by notL 1
| ...
| ⊢ []=[]
|                                                          by equality √
|\
| 1. x:int
| ⊢ ∼([x]=[]) ⇒ ∃i:int. (i∈[x] ∧ ∀j:int. (j∈[x] ⇒ j≤i))
| |                                                        by impR
| ...
| 2. ∼([x]=[])
| ⊢ ∃i:int. (i∈[x] ∧ ∀j:int. (j∈[x] ⇒ j≤i))
| |                                                        by exR x
| ...
| ⊢ x∈[x] ∧ ∀j:int. (j∈[x] ⇒ j≤x)
| |                                                        by andR
| |\
| | ...
| | ⊢ x∈[x]
| | |                                                      by unfold ∈ in 0
| | ...
| | ⊢ ∃k:int. (index(k,[x])=i ∧ 0≤k ∧ k<length([x]))
| | |                                                      by exR 0
| | ...
| | ⊢ index(0,[x])=x ∧ 0≤0 ∧ 0<length([x])
| | |                                                      by simplify in 0
| | ...
| | ⊢ x=x ∧ 0≤0 ∧ 0<1| |                                   by arith √
|  \
|   ...
|   ⊢ ∀j:int. (j∈[x] ⇒ j≤x)
|   |                                                      by allR
|   ...
|   3. j:int
|   ⊢ j∈[x] ⇒ j≤x
|   |                                                      by impR
|   |
|   ...
|   4. j∈[x]
|   ⊢ j≤x
|   |                                                      by unfold ∈ in 4
|   ...
|   4. ∃k:int. (index(k,[x])=j ∧ 0≤k ∧ k<length([x]))
|   ⊢ j≤x
|   |                                                      by exL 4
```

```
|   |
|   ...
|   4. k:int
|   5. index(k,[x])=j ∧ 0≤k ∧ k<length([x])
|   ⊢ j≤x
|   |                                                          by simplify in 5
|   ...
|   5. index(k,[x])=j ∧ 0≤k ∧ k<1
|   ⊢ j≤x
|   |                                                          by andL 5
|   ...
|   5. index(k,[x])=j
|   6. 0≤k
|   7. k<1
|   ⊢ j≤x
|   |                                                          by cut k=0
|   |\
|   | ...
|   | ⊢ k=0
|   |                                                          by arith √
|    \
|     ...
|     8. k=0
|     ⊢ j≤x
|     |                                                        by cut index(k,[x])=j
|     |\
|     | ...
|     | ⊢ index(k,[x])=j
|     |                                                        by hyp 5 √
|      \
|       ...
|       9. index(k,[x])=j
|       ⊢ j≤x
|       |                                                      by subst 8 9
|       ...
|       9. index(0,[x])=j
|       ⊢ j≤x
|       |                                                      by simplify 9
|       ...
|       9. x=j
|       ⊢ j≤x
|                                                              by arith √
 \
  1. x:int
  2. l:list
  3. ∼(l=[]) ⇒ ∃i:int. (i∈l ∧ ∀j:int. (j∈l ⇒ j≤i))
  ⊢ ∼(x::l=[]) ⇒ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
  |                                                            by cut (l=[]) ∨ ∼(l=[])
  |\
  | ...
  | ⊢ (l=[]) ∨ ∼(l=[])
  |                                                            by equality √
   \
    ...
    4. (l=[]) ∨ ∼(l=[])
    ⊢ ∼(x::l=[]) ⇒ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
    |                                                          by orL 4
    |\
    | ...
    | 4. l=[]
    | ⊢ ∼(x::l=[]) ⇒ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
    | |                                                        by subst 4 0
    | ...
    | ⊢ ∃i:int. (i∈x::[] ∧ ∀j:int. (j∈x::[] ⇒ j≤i))
    | |                                                        by simplify 0
```

```
| |
| ...
| ⊢ ∃i:int. (i∈[x] ∧ ∀j:int. (j∈[x] ⇒ j≤i))
|                                                                    by proof of previous case √
|
 \
  ...
  4. ∼(l=[])
  ⊢ ∼(x::l=[]) ⇒ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
  |                                                                  by impR
  ...
  5. ∼(x::l=[])
  ⊢ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
  |                                                                  by impL 3
  |\
  | ...
  | ⊢ ∼(l=[])
  |                                                                  by hyp 4 √
   \
    ...
    3. ∼(l=[])
    4. ∼(x::l=[])
    5. ∼(l=[])
    6. ∃i:int. (i∈l ∧ ∀j:int. (j∈l ⇒ j≤i))
    ⊢ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
    |                                                                by exL 6
    ...
    6. i:int
    7. i∈l ∧ ∀j:int. (j∈l ⇒ j≤i)
    ⊢ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
    |                                                                by andL 7
    ...
    7. i∈l
    8. ∀j:int. (j∈l ⇒ j≤i)
    ⊢ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
    |                                                                by unfold ∈ 7
    ...
    7. ∃k:int. (index(k,l)=i ∧ 0≤k ∧ k<length(x::l))
    8. ∀j:int. (j∈l ⇒ j≤i)
    ⊢ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
    |                                                                by exL 7
    ...
    7. k:int
    8. index(k,l)=i ∧ 0≤k ∧ k<length(x::l)
    9. ∀j:int. (j∈l ⇒ j≤i)
    ⊢ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
    |                                                                by andL 8
    ...
    8. index(k,l)=i
    9. 0≤k
    10. k<length(x::l)
    11. ∀j:int. (j∈l ⇒ j≤i)
    ⊢ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
    |                                                                by simplify 10
    ...
    8. index(k,l)=i
    9. 0≤k
    10. k<length(l)+1
    11. ∀j:int. (j∈l ⇒ j≤i)
    ⊢ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
    |                                                                by cut (x ≤ i) ∨ (x > i)
    |\
    | ...
    | ⊢ (x ≤ i) ∨ (x > i)
    |                                                                by arith √
```

```
|
 \
  ...
  12. (x ≤ i) ∨ (x > i)
  ⊢ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
  |                                                          by orL 12
  |\
  | ...
  | 12. x ≤ i
  | ⊢ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
  | |                                                        by exR i
  | ...
  | ⊢ i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i)
  | |                                                        by andR
  | |\
  | | ...
  | | ⊢ i∈x::l
  | | |                                                      by unfold ∈ in 0
  | | ...
  | | ⊢ ∃k:int. (index(k,x::l)=i ∧ 0≤k ∧ k<length(x::l))
  | | |                                                      by exR k+1
  | | ...
  | | ⊢ index(k+1,x::l)=i ∧ 0≤k+1 ∧ k+1<length(x::l)
  | | |                                                      by def index(k+1,x::l) up
  | | |\
  | | | ...
  | | | ⊢ 0 < k+1
  | | |                                                      by arith √
  | |  \
  | |   ...
  | |   13. 0 < k+1
  | |   14. index(k+1,x::l) = index(k+1-1,tl(x::l))
  | |   ⊢ index(k+1,x::l)=i ∧ 0≤k+1 ∧ k+1<length(x::l)
  | |   |                                                    by subst 14 0
  | |   ...
  | |   ⊢ index(k+1-1,tl(x::l))=i ∧ 0≤k+1 ∧ k+1<length(x::l)
  | |   |                                                    by simplify 0
  | |   ...
  | |   ⊢ index(k,l)=i ∧ 0≤k+1 ∧ k+1<length(l)+1
  | |                                                        by arith √
  |  \
  |   ...
  |   ⊢ ∀j:int. (j∈x::l ⇒ j≤i)
  |   |                                                      by allR
  |   ...
  |   13. j: int
  |   ⊢ j∈x::l ⇒ j≤i
  |   |                                                      by impR
  |   ...
  |   14. j∈x::l
  |   ⊢ j≤i
  |   |                                                      by unfold ∈ in 14
  |   ...
  |   14. ∃k:int. (index(k,x::l)=j ∧ 0≤k ∧ k<length(x::l))
  |   ⊢ j≤i
  |   |                                                      exL 14
  |   ...
  |   14. k':int
  |   15. index(k',x::l)=j ∧ 0≤k' ∧ k'<length(x::l))
  |   ⊢ j≤i
  |   |                                                      andL 15
```

```
|   |
|   ...
|   15. index(k',x::l)=j
|   16. 0≤k'
|   17. k'<length(x::l))
|   ⊢ j≤i
|   |                                                    by simplify 17
|   ...
|   17. k'<length(l)+1
|   ⊢ j≤i
|   |                                                    cut k'=0 ∨ k'>0
|   |\
|   | ...
|   | ⊢ k'=0 ∨ k'>0
|   |                                                    by arith √
|    \
|     ...
|     18. k'=0 ∨ k'>0
|     ⊢ j≤i
|     |                                                  by orL 18
|     |\
|     | ...
|     | 18. k'=0
|     | ⊢ j≤i
|     | |                                                by cut index(k',x::l)=j
|     | |\
|     | | ...
|     | | ⊢ index(k',x::l)=j
|     | |                                                by hyp 15
|     |  \
|     |   ...
|     |   19. index(k',x::l)=j
|     |   ⊢ j≤i
|     |   |                                              by subst 18 19
|     |   ...
|     |   19. index(0,x::l)=j
|     |   ⊢ j≤i
|     |   |                                              by simplify 19
|     |   ...
|     |   19. x=j
|     |   ⊢ j≤i
|     |                                                  by arith √
|      \
|       ...
|       18. k'>0
|       ⊢ j≤i
|       |                                                by allL 11 j
|       ...
|       19. j∈l ⇒ j≤i
|       ⊢ j≤i
|       |                                                by impL 19
|       |\
|       | ...
|       | ⊢ j∈l
|       | |                                              by unfold ∈ in 0
|       | ...
|       | ⊢ ∃k:int. (index(k,l)=j ∧ 0≤k ∧ k<length(l))
|       | |                                              by exR k'-1
|       | ...
|       | ⊢ index(k'-1,l)=j ∧ 0≤k'-1 ∧ k'-1<length(l)
|       | |                                              by def index(k',x::l) up
|       | |\
|       | | ...
|       | | ⊢ 0≤k'
|       | |                                              by hyp 18
```

```
|       | |
|       |  \
|       |   ...
|       |   20. 0≤k'
|       |   21. index(k',x::l) = index(k'-1,tl(x::l))
|       |   ⊢ index(k'-1,l)=j ∧ 0≤k'-1 ∧ k'-1<length(l)
|       |   |                                                  by simplify 21
|       |   ...
|       |   21. index(k',x::l) = index(k'-1,l)
|       |   ⊢ index(k'-1,l)=j ∧ 0≤k'-1 ∧ k'-1<length(l)
|       |                                                      by arith √
|       |\
|       | 19. j∈l
|       | 20. j≤i
|       | ⊢ j≤i
|                                                              by hyp 20 √
|\
| ...
| 12. x>i
| ⊢ ∃i:int. (i∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤i))
| |                                                            by exR x
| ...
| ⊢ x∈x::l ∧ ∀j:int. (j∈x::l ⇒ j≤x)
| |                                                            by andR
| |\
| | ...
| | ⊢ x∈x::l
| | |                                                          by unfold ∈ in 0
| | ...
| | ⊢ ∃k:int. (index(k,x::l)=x ∧ 0≤k ∧ k<length(x::l))
| | |                                                          by exR 0
| | ...
| | ⊢ index(0,x::l)=x ∧ 0≤0 ∧ 0<length(x::l)
| | |                                                          by simplify 0
| | ...
| | ⊢ x=x ∧ 0≤0 ∧ 0<length(l)+1
| |                                                            by arith √
| |\
| | ...
| | ⊢ ∀j:int. (j∈x::l ⇒ j≤x)
| | |                                                          by allR
| | ...
| | 13. j:int
| | ⊢ j∈x::l ⇒ j≤x
| | |                                                          by impR
| | ...
| | 14. j∈x::l
| | ⊢ j≤x
| | |                                                          by unfold ∈ in 14
| | ...
| | 14. ∃k:int. (index(k,x::l)=j ∧ 0≤k ∧ k<length(x::l))
| | ⊢ j≤x
| | |                                                          by exL 14
| | ...
| | 14. k':int
| | 15. index(k',x::l)=j ∧ 0≤k' ∧ k'<length(x::l)
| | ⊢ j≤x
| | |                                                          by andL 15
| | ...
| | 15. index(k',x::l)=j
| | 16. 0≤k'
| | 17. k'<length(x::l)
| | ⊢ j≤x
| | |                                                          by simplify 17
```

```
|
...
17. k'<length(l)+1
⊢ j≤x
|                                                    by cut k'=0 ∨ k'>0
|\
| ...
| ⊢ k'=0 ∨ k'>0
|                                                    by arith √
 \
  ...
  18. k'=0 ∨ k'>0
  ⊢ j≤x
  |                                                  by orL 18
  |\
  | ...
  | 18. k'=0
  | ⊢ j≤x
  | |                                                by cut index(k',x::l)=j
  | |\
  | | ...
  | | ⊢ index(k',x::l)=j
  | |                                                by hyp 15
  |  \
  |   ...
  |   19. index(k',x::l)=j
  |   ⊢ j≤x
  |   |                                              by subst 18 19
  |   ...
  |   19. index(0,x::l)=j
  |   ⊢ j≤x
  |   |                                              by simplify 19
  |   ...
  |   19. x=j
  |   ⊢ j≤x
  |                                                  by arith √
   \
    ...
    18. k'>0
    ⊢ j≤x
    |                                                by allL 11 j
    ...
    19. j∈l ⇒ j≤i
    ⊢ j≤x
    |                                                by impL 19
    |\
    | ...
    | ⊢ j∈l
    | |                                              by unfold ∈ in 0
    | ...
    | ⊢ ∃k:int. (index(k,l)=j ∧ 0≤k ∧ k<length(l))
    | |                                              exR k'-1
    | ...
    | ⊢ index(k'-1,l)=j ∧ 0≤k'-1 ∧ k'-1<length(l))
    | |                                              by def index(k',x::l) up
    | |\
    | | ...
    | | ⊢ 0≤k'
    | |                                              by hyp 18
    |  \
    |   ...
    |   20. 0≤k'
    |   21. index(k',x::l) = index(k'-1,tl(x::l))
    |   ⊢ index(k'-1,l)=j ∧ 0≤k'-1 ∧ k'-1<length(l))
    |   |                                            by simplify 21
```

```
|   |
|   ...
|   20. 0≤k'
|   21. index(k',x::l) = index(k'-1,l)
|   ⊢ index(k'-1,l)=j ∧ 0≤k'-1 ∧ k'-1<length(l))
|                                                    by arith √
 \
  ...
  19. j∈l
  20. j≤i
  ⊢ j≤x
                                                     by arith √
```

We describe the extracted algorithm in λ-PRL notation.

```
listmax(l:list):int ≡  0 => 0
                       1 => hd(l)
                       l => if tl(l) = []
                              then hd(l)
                              else if hd(l) ≤ listmax(tl(l))
                                     then listmax(tl(l))
                                     else hd(l)
```