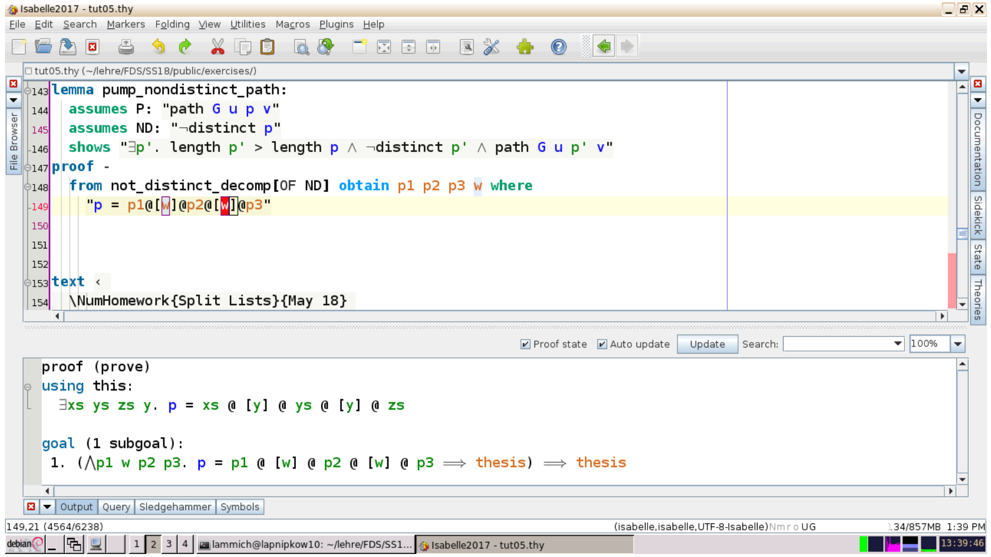
Script generated by TTT

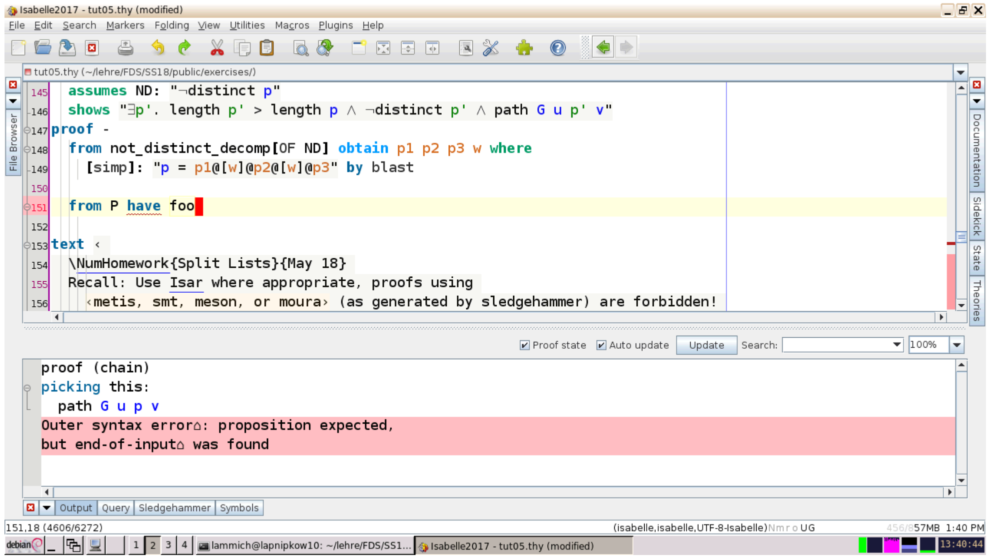
Title: Lammich: FDS Tutorial (11.05.2018)

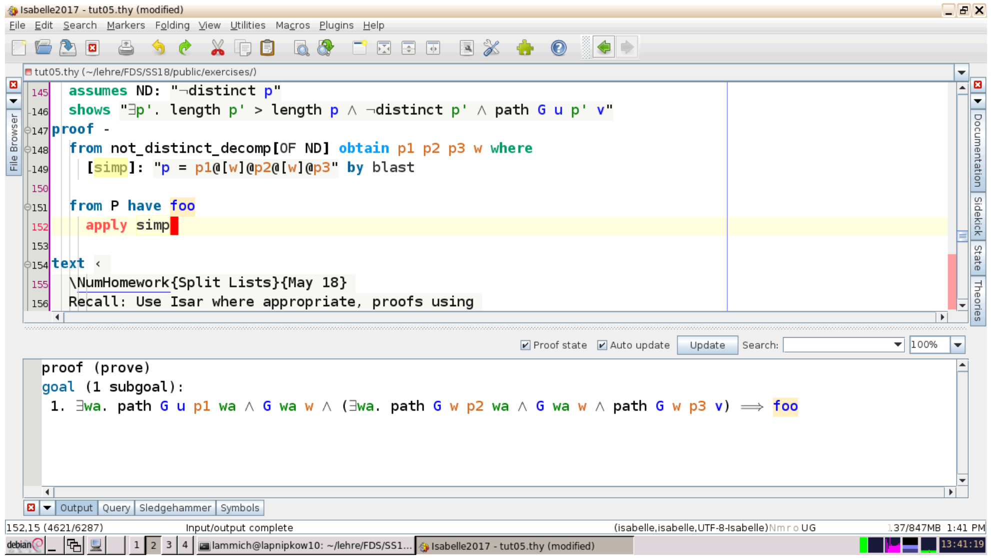
Date: Fri May 11 13:39:45 CEST 2018

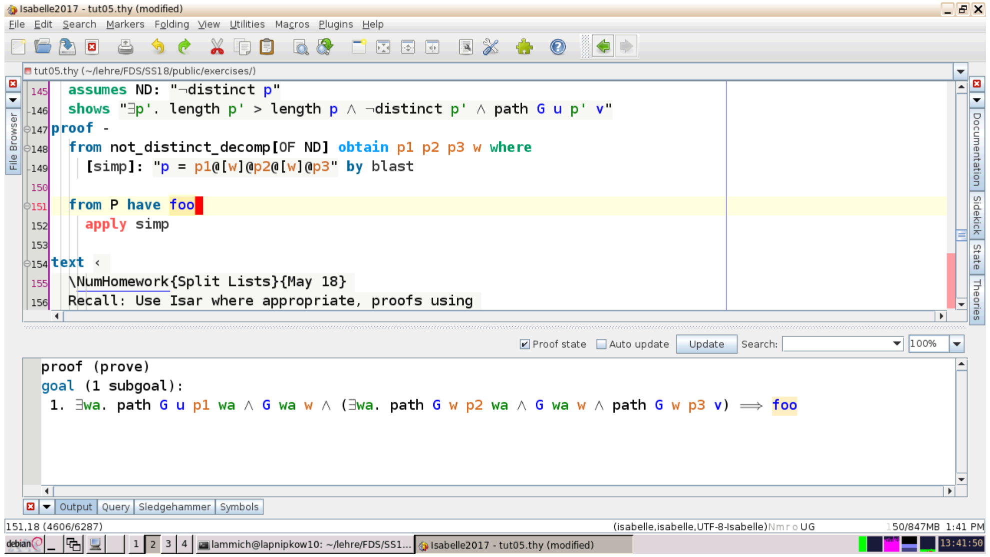
Duration: 13:44 min

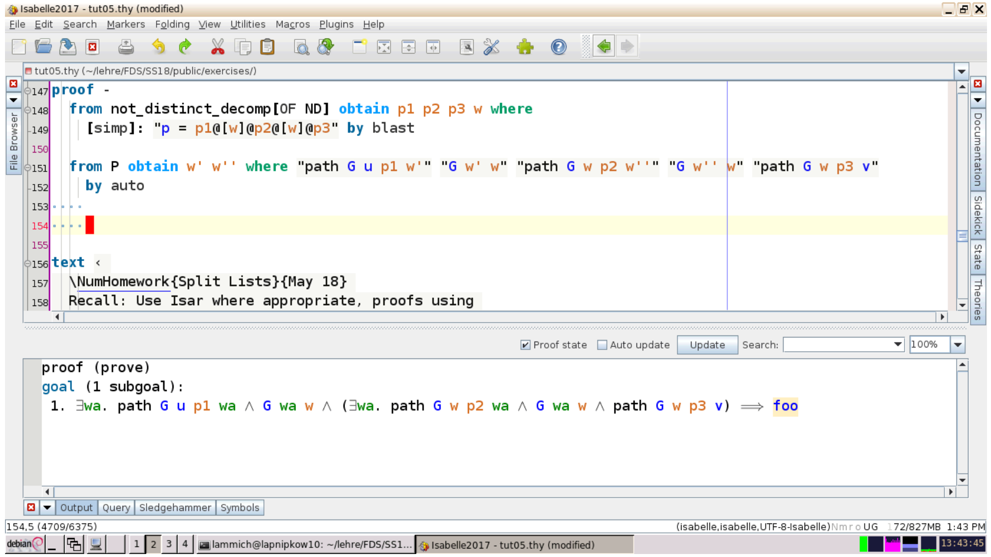
Pages: 22

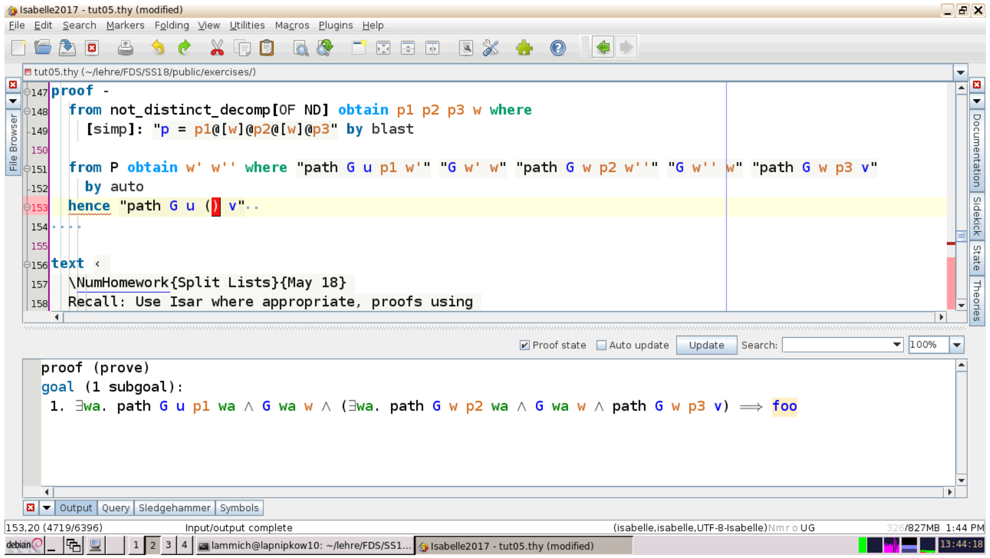


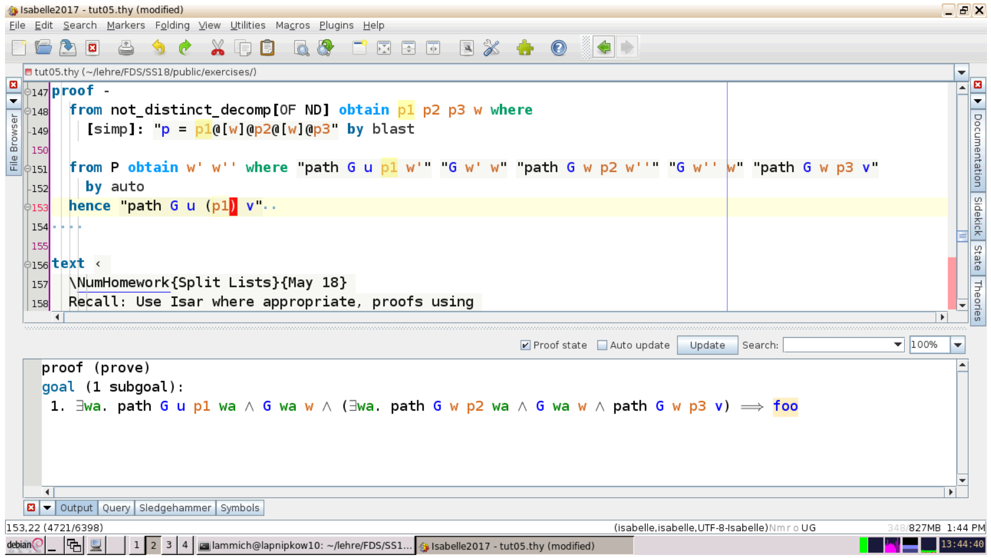


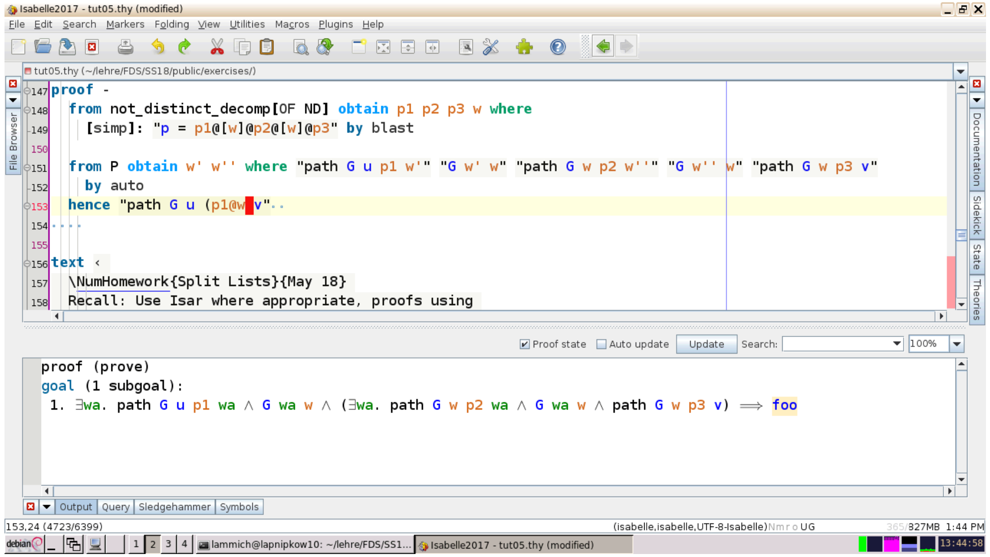


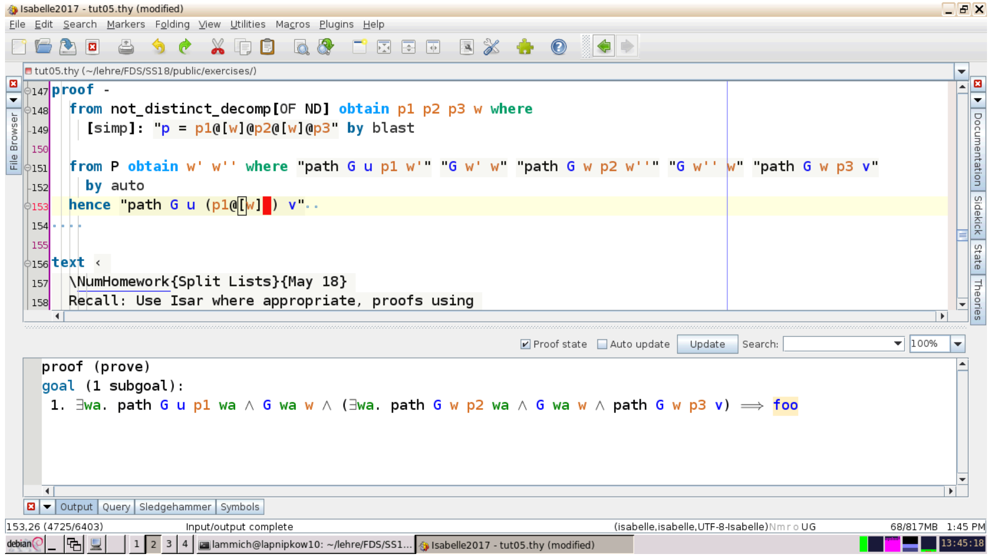

Isabelle2017 - tut05.thy (modified)
assumes ND: "¬distinct p"
shows "∃p'. length p' > length p ∧ ¬distinct p' ∧ path G u p' v"
proof -
from not_distinct_decomp[OF ND] obtain p1 p2 p3 w where
[simp]: "p = p1@[w]@p2@[w]@p3" by blast
from P have foo
apply simp
text ‹
\NumHomework{Split Lists}{May 18}
Recall: Use Isar where appropriate, proofs using
proof (prove)
goal (1 subgoal):
1. ∃wa. path G u p1 wa ∧ G wa w ∧ (∃wa. path G w p2 wa ∧ G wa w ∧ path G w p3 v) ⟹ foo

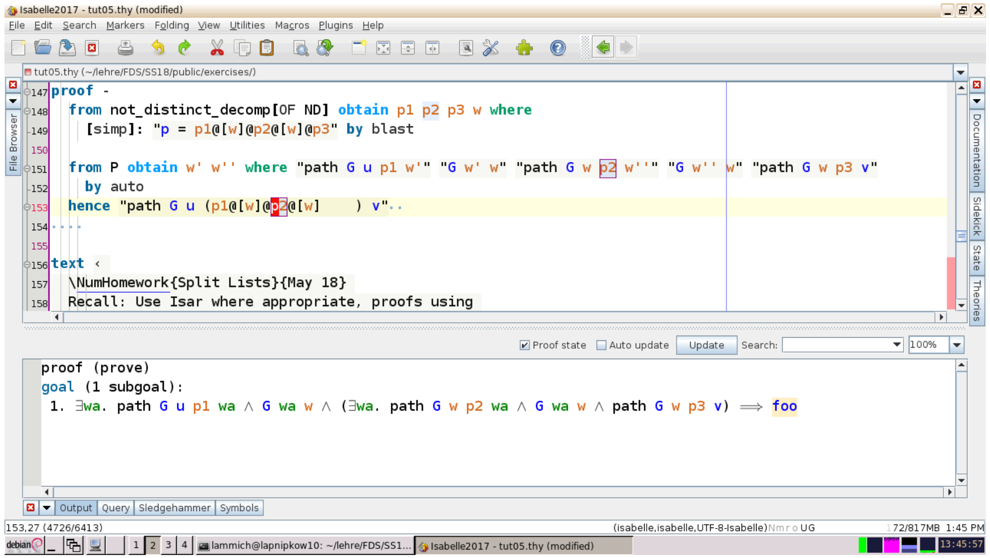

Isabelle2017 - tut05.thy (modified)
proof -
from not_distinct_decomp[OF ND] obtain p1 p2 p3 w where
[simp]: "p = p1@[w]@p2@[w]@p3" by blast
from P obtain w' w'' where "path G u p1 w'" "G w' w" "path G w p2 w''" "G w'' w" "path G w p3 v"
by auto
text ‹
\NumHomework{Split Lists}{May 18}
Recall: Use Isar where appropriate, proofs using
proof (prove)
goal (1 subgoal):
1. ∃wa. path G u p1 wa ∧ G wa w ∧ (∃wa. path G w p2 wa ∧ G wa w ∧ path G w p3 v) ⟹ foo

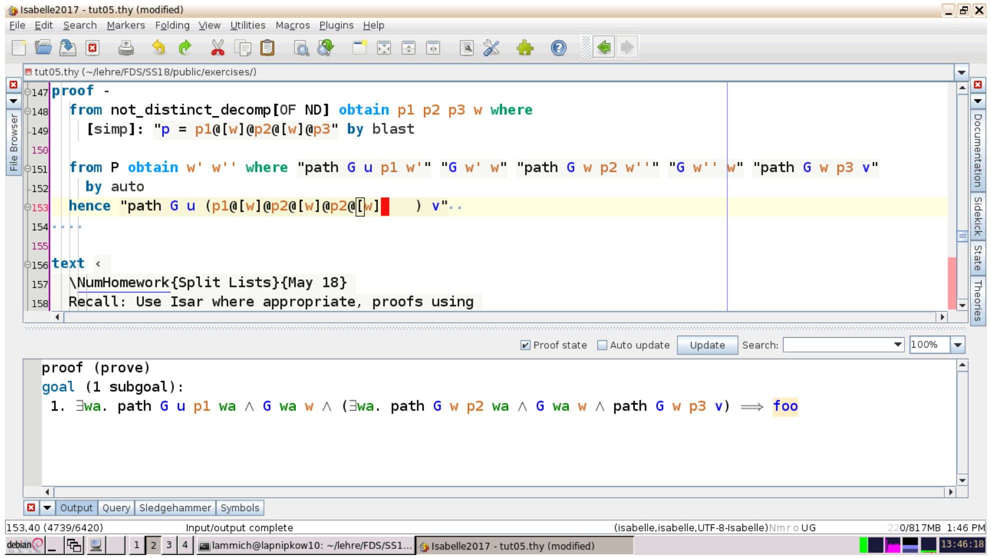

Isabelle2017 - tut05.thy (modified)
proof -
from not_distinct_decomp[OF ND] obtain p1 p2 p3 w where
[simp]: "p = p1@[w]@p2@[w]@p3" by blast
from P obtain w' w'' where "path G u p1 w'" "G w' w" "path G w p2 w''" "G w'' w" "path G w p3 v"
by auto
hence "path G u () v"
text ‹
\NumHomework{Split Lists}{May 18}
Recall: Use Isar where appropriate, proofs using
proof (prove)
goal (1 subgoal):
1. ∃wa. path G u p1 wa ∧ G wa w ∧ (∃wa. path G w p2 wa ∧ G wa w ∧ path G w p3 v) ⟹ foo

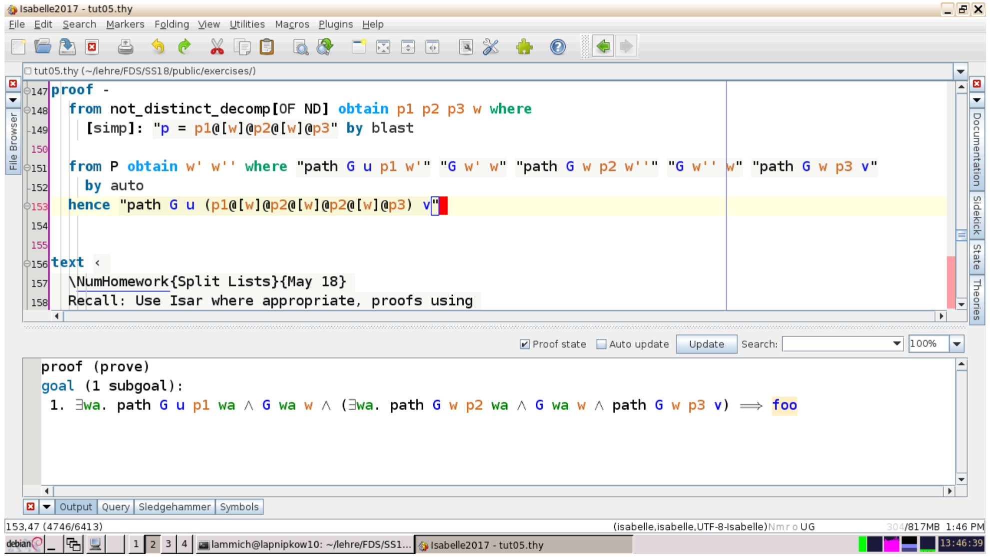

Isabelle2017 - tut05.thy (modified)
proof -
from not_distinct_decomp[OF ND] obtain p1 p2 p3 w where
[simp]: "p = p1@[w]@p2@[w]@p3" by blast
from P obtain w' w'' where "path G u p1 w'" "G w' w" "path G w p2 w''" "G w'' w" "path G w p3 v"
by auto
hence "path G u (p1) v"
text ‹
\NumHomework{Split Lists}{May 18}
Recall: Use Isar where appropriate, proofs using
proof (prove)
goal (1 subgoal):
1. ∃wa. path G u p1 wa ∧ G wa w ∧ (∃wa. path G w p2 wa ∧ G wa w ∧ path G w p3 v) ⟹ foo

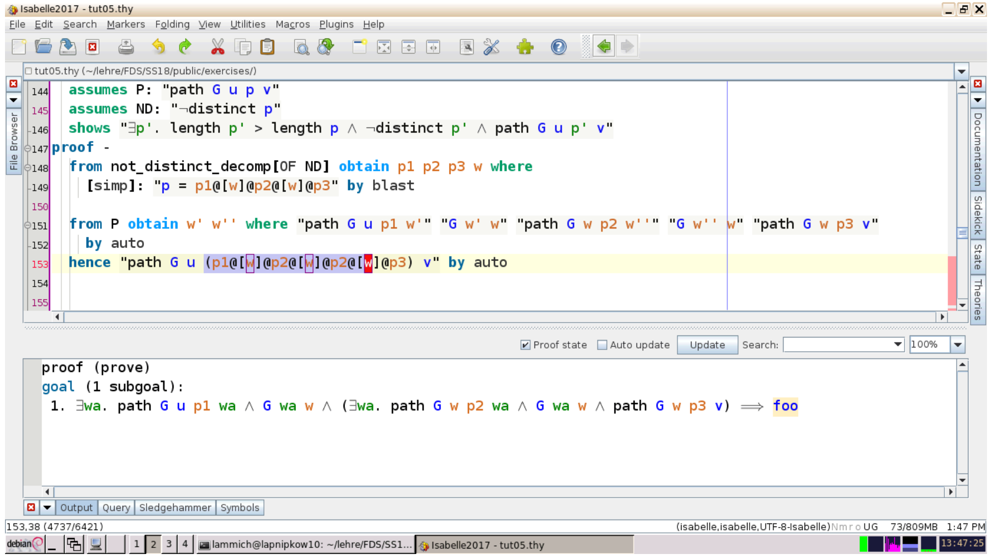

Isabelle2017 - tut05.thy (modified)
tut05.thy (~/lehre/FDS/SS18/public/exercises/)
proof -
from not_distinct_decomp[OF ND] obtain p1 p2 p3 w where
[simp]: "p = p1@[w]@p2@[w]@p3" by blast
from P obtain w' w'' where "path G u p1 w'" "G w' w" "path G w p2 w''" "G w'' w" "path G w p3 v"
by auto
hence "path G u (p1@w v"..
text ‹
\NumHomework{Split Lists}{May 18}
Recall: Use Isar where appropriate, proofs using
proof (prove)
goal (1 subgoal):
1. ∃wa. path G u p1 wa ∧ G wa w ∧ (∃wa. path G w p2 wa ∧ G wa w ∧ path G w p3 v) ⟹ foo

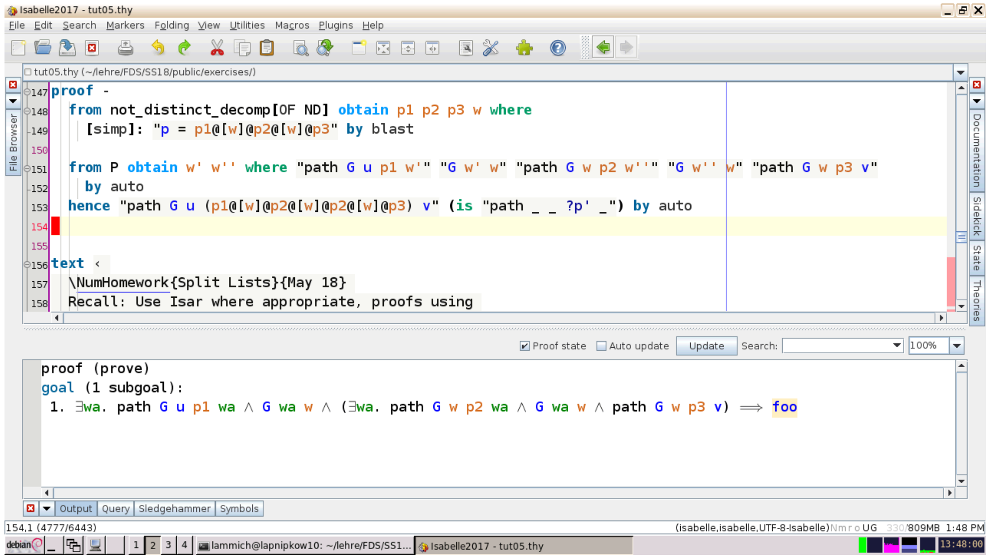

Isabelle2017 - tut05.thy (modified)
tut05.thy (~/lehre/FDS/SS18/public/exercises/)
proof -
from not_distinct_decomp[OF ND] obtain p1 p2 p3 w where
[simp]: "p = p1@[w]@p2@[w]@p3" by blast
from P obtain w' w'' where "path G u p1 w'" "G w' w" "path G w p2 w''" "G w'' w" "path G w p3 v"
by auto
hence "path G u (p1@[w]) v"..
text ‹
\NumHomework{Split Lists}{May 18}
Recall: Use Isar where appropriate, proofs using
proof (prove)
goal (1 subgoal):
1. ∃wa. path G u p1 wa ∧ G wa w ∧ (∃wa. path G w p2 wa ∧ G wa w ∧ path G w p3 v) ⟹ foo

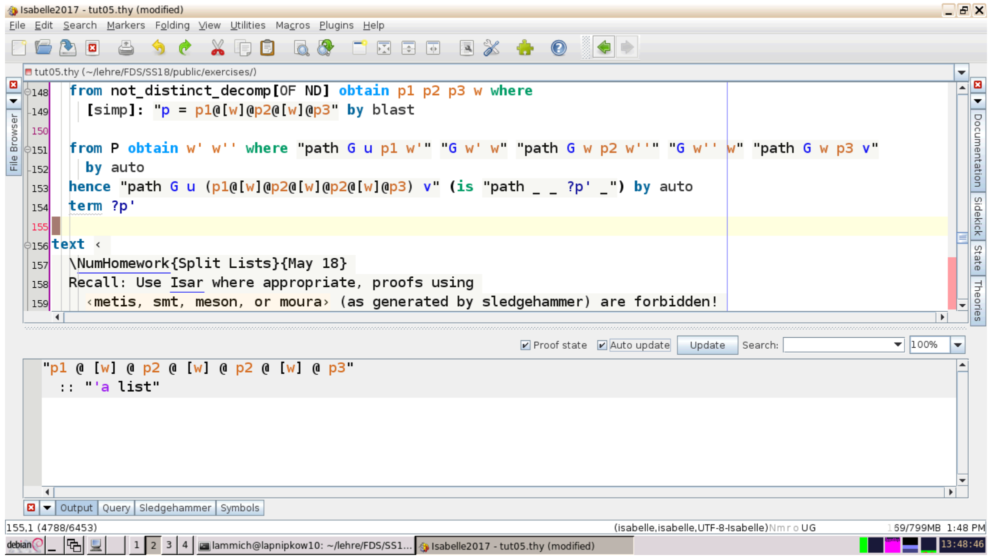

Isabelle2017 - tut05.thy (modified)
tut05.thy (~/lehre/FDS/SS18/public/exercises/)
proof -
from not_distinct_decomp[OF ND] obtain p1 p2 p3 w where
[simp]: "p = p1@[w]@p2@[w]@p3" by blast
from P obtain w' w'' where "path G u p1 w'" "G w' w" "path G w p2 w''" "G w'' w" "path G w p3 v"
by auto
hence "path G u (p1@[w]@p2@[w]) v"..
text ‹
\NumHomework{Split Lists}{May 18}
Recall: Use Isar where appropriate, proofs using
proof (prove)
goal (1 subgoal):
1. ∃wa. path G u p1 wa ∧ G wa w ∧ (∃wa. path G w p2 wa ∧ G wa w ∧ path G w p3 v) ⟹ foo

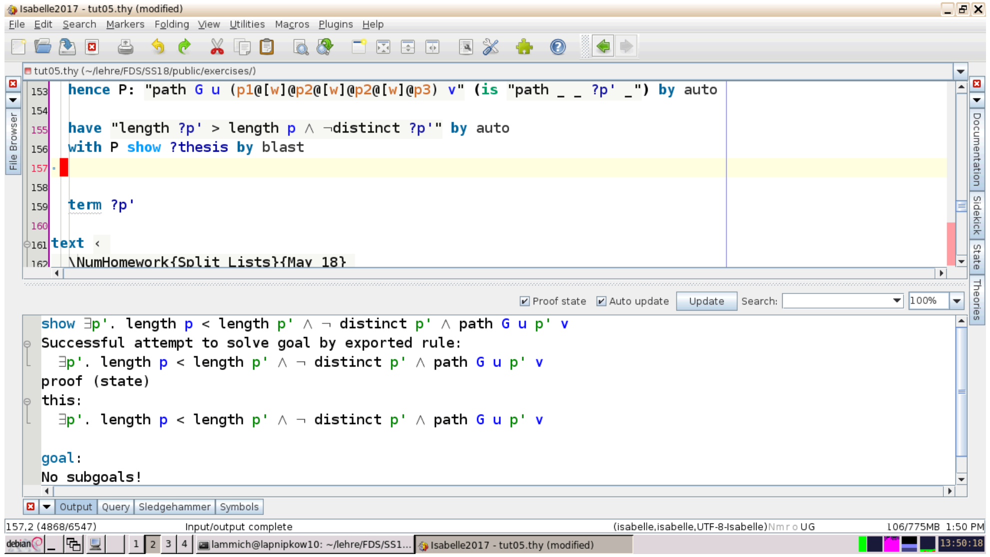

Isabelle2017 - tut05.thy (modified)
tut05.thy (~/lehre/FDS/SS18/public/exercises/)
proof -
from not_distinct_decomp[OF ND] obtain p1 p2 p3 w where
[simp]: "p = p1@[w]@p2@[w]@p3" by blast
from P obtain w' w'' where "path G u p1 w'" "G w' w" "path G w p2 w''" "G w'' w" "path G w p3 v"
by auto
hence "path G u (p1@[w]@p2@[w]@p2@[w]) v"..
text ‹
\NumHomework{Split Lists}{May 18}
Recall: Use Isar where appropriate, proofs using
proof (prove)
goal (1 subgoal):
1. ∃wa. path G u p1 wa ∧ G wa w ∧ (∃wa. path G w p2 wa ∧ G wa w ∧ path G w p3 v) ⟹ foo

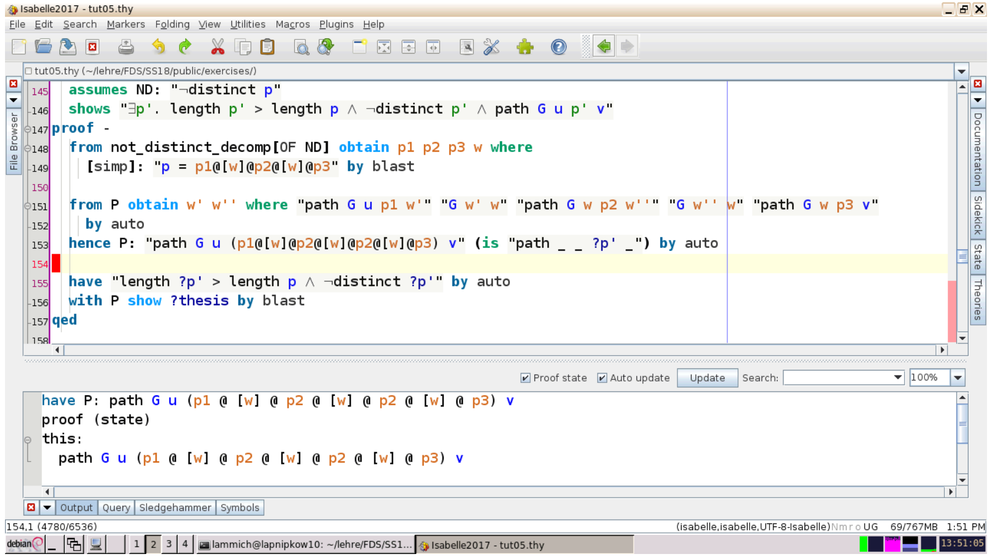

Isabelle2017 - tut05.thy
proof -
from not_distinct_decomp[OF ND] obtain p1 p2 p3 w where
[simp]: "p = p1@[w]@p2@[w]@p3" by blast
from P obtain w' w'' where "path G u p1 w''" "G w' w" "path G w p2 w'''" "G w'' w" "path G w p3 v"
by auto
hence "path G u (p1@[w]@p2@[w]@p2@[w]@p3) v"
text ‹
\NumHomework{Split Lists}{May 18}
Recall: Use Isar where appropriate, proofs using
proof (prove)
goal (1 subgoal):
1. ∃wa. path G u p1 wa ∧ G wa w ∧ (∃wa. path G w p2 wa ∧ G wa w ∧ path G w p3 v) ⟹ foo

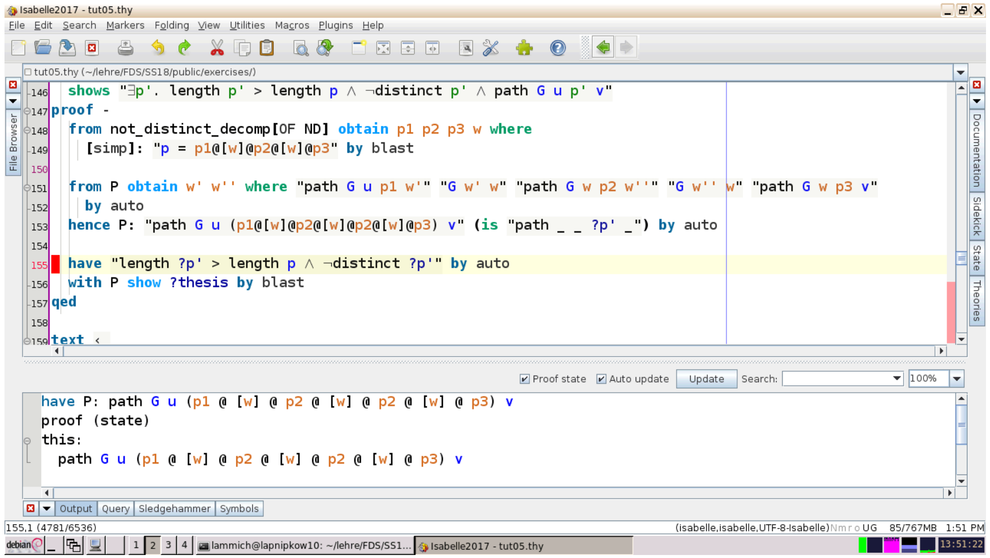

Isabelle2017 - tut05.thy
assumes P: "path G u p v"
assumes ND: "¬distinct p"
shows "∃p'. length p' > length p ∧ ¬distinct p' ∧ path G u p' v"
proof -
from not_distinct_decomp[OF ND] obtain p1 p2 p3 w where
[simp]: "p = p1@[w]@p2@[w]@p3" by blast
from P obtain w' w'' where "path G u p1 w''" "G w' w" "path G w p2 w'''" "G w'' w" "path G w p3 v"
by auto
hence "path G u (p1@[w]@p2@[w]@p2@[w]@p3) v" by auto
proof (prove)
goal (1 subgoal):
1. ∃wa. path G u p1 wa ∧ G wa w ∧ (∃wa. path G w p2 wa ∧ G wa w ∧ path G w p3 v) ⟹ foo

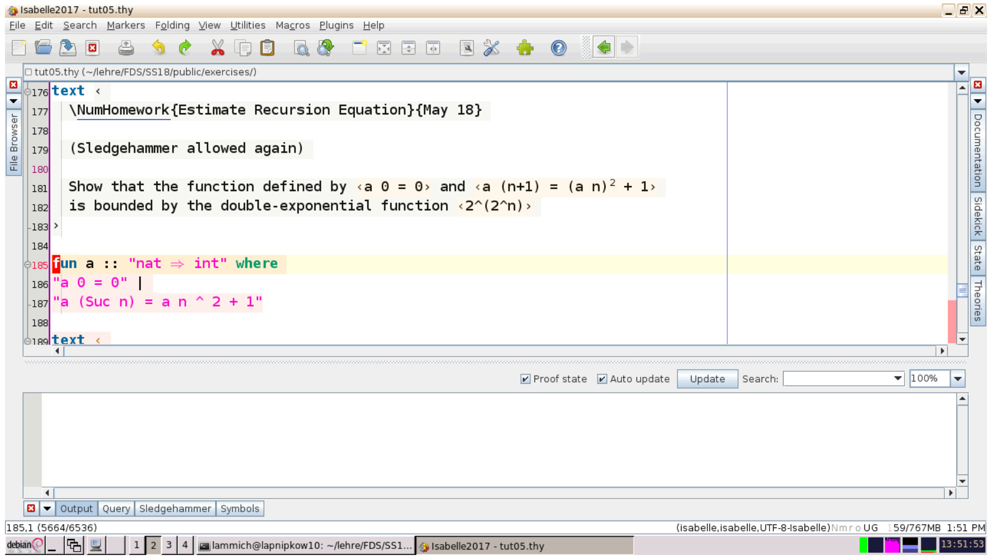

Isabelle2017 - tut05.thy
proof -
from not_distinct_decomp[OF ND] obtain p1 p2 p3 w where
[simp]: "p = p1@[w]@p2@[w]@p3" by blast
from P obtain w' w'' where "path G u p1 w''" "G w' w" "path G w p2 w'''" "G w'' w" "path G w p3 v"
by auto
hence "path G u (p1@[w]@p2@[w]@p2@[w]@p3) v" (is "path _ _ ?p' _") by auto
text ‹
\NumHomework{Split Lists}{May 18}
Recall: Use Isar where appropriate, proofs using
proof (prove)
goal (1 subgoal):
1. ∃wa. path G u p1 wa ∧ G wa w ∧ (∃wa. path G w p2 wa ∧ G wa w ∧ path G w p3 v) ⟹ foo

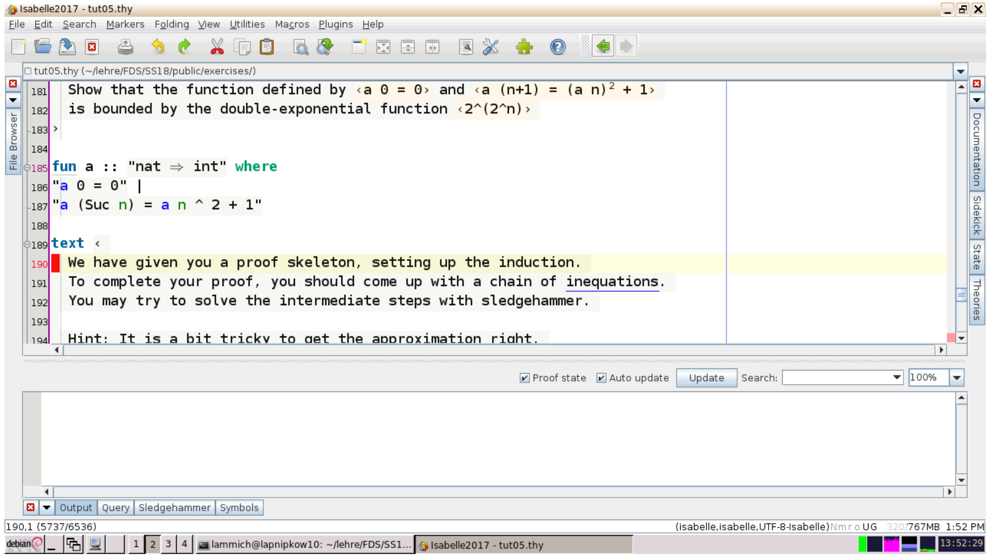

Isabelle2017 - tut05.thy (modified)
from not_distinct_decomp[OF ND] obtain p1 p2 p3 w where
[simp]: "p = p1@[w]@p2@[w]@p3" by blast
from P obtain w' w'' where "path G u p1 w''" "G w' w" "path G w p2 w'''" "G w'' w" "path G w p3 v"
by auto
hence "path G u (p1@[w]@p2@[w]@p2@[w]@p3) v" (is "path _ _ ?p' _") by auto
term ?p'
text ‹
\NumHomework{Split Lists}{May 18}
Recall: Use Isar where appropriate, proofs using
‹metis, smt, meson, or moura› (as generated by sledgehammer) are forbidden!
"p1 @ [w] @ p2 @ [w] @ p2 @ [w] @ p3"
:: "'a list"

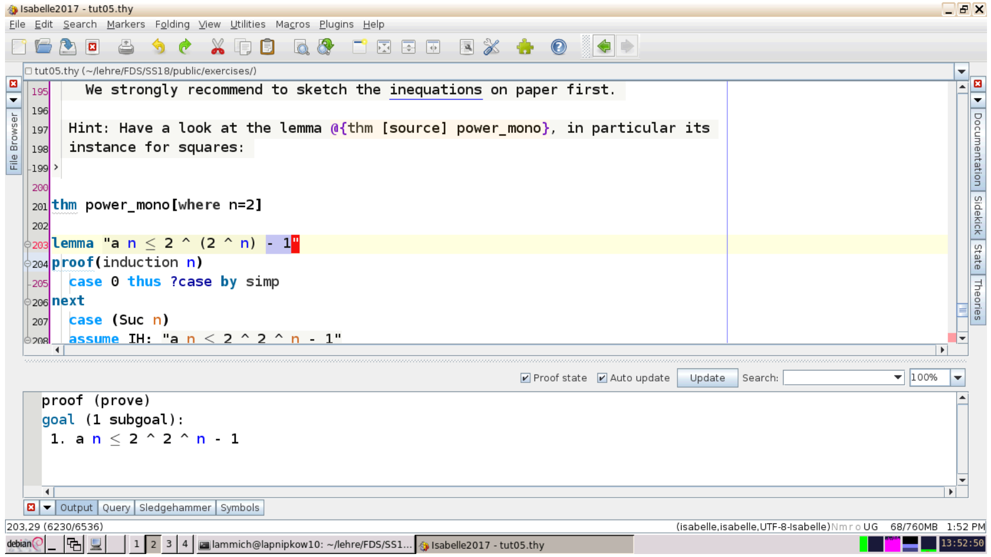

Isabelle2017 - tut05.thy (modified)
File Edit Search Markers Folding View Utilities Macros Plugins Help
tut05.thy (~/lehre/FDS/SS18/public/exercises/)
hence P: "path G u (p1@[w]@p2@[w]@p2@[w]@p3) v" (is "path _ _ ?p' _") by auto
have "length ?p' > length p ∧ ¬distinct ?p'" by auto
with P show ?thesis by blast
term ?p'
text ‹
\NumHomework{Split Lists}{May 18}
Proof state
Auto update
Update
Search:
100%
show ∃p'. length p < length p' ∧ ¬ distinct p' ∧ path G u p' v
Successful attempt to solve goal by exported rule:
∃p'. length p < length p' ∧ ¬ distinct p' ∧ path G u p' v
proof (state)
this:
∃p'. length p < length p' ∧ ¬ distinct p' ∧ path G u p' v
goal:
No subgoals!
Output
Query
Sledgehammer
Symbols
157,2 (4868/6547)
Input/output complete
(isabelle,isabelle,UTF-8-Isabelle)
106/775MB 1:50 PM
lammich@lapnipkow10: ~/lehre/FDS/SS1...
Isabelle2017 - tut05.thy (modified)
13:50:18

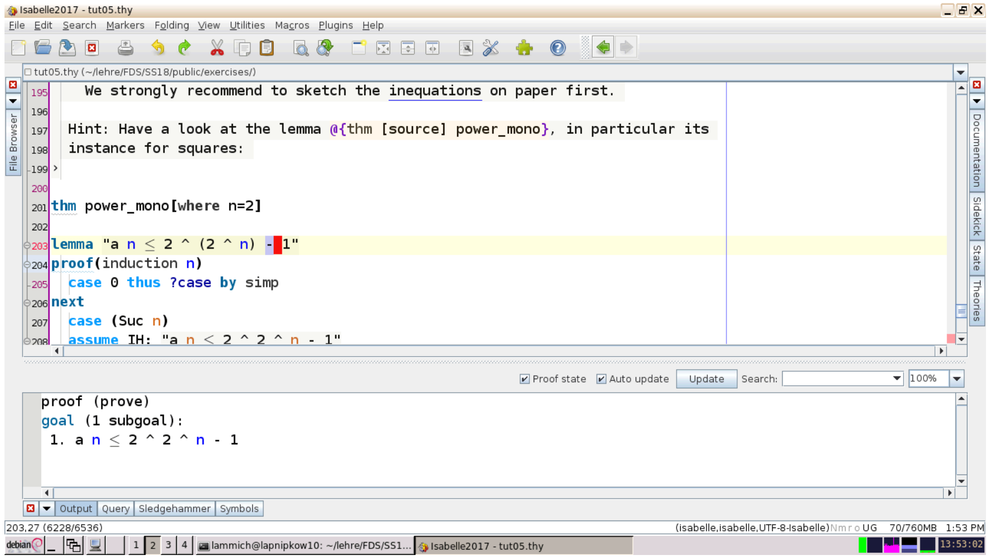

Isabelle2017 - tut05.thy
File Edit Search Markers Folding View Utilities Macros Plugins Help
tut05.thy (~/lehre/FDS/SS18/public/exercises/)
assumes ND: "¬distinct p"
shows "∃p'. length p' > length p ∧ ¬distinct p' ∧ path G u p' v"
proof -
from not_distinct_decomp[OF ND] obtain p1 p2 p3 w where
[simp]: "p = p1@[w]@p2@[w]@p3" by blast
from P obtain w' w'' where "path G u p1 w'" "G w' w" "path G w p2 w''" "G w'' w" "path G w p3 v"
by auto
hence P: "path G u (p1@[w]@p2@[w]@p2@[w]@p3) v" (is "path _ _ ?p' _") by auto
have "length ?p' > length p ∧ ¬distinct ?p'" by auto
with P show ?thesis by blast
qed
Proof state
Auto update
Update
Search:
100%
have P: path G u (p1 @ [w] @ p2 @ [w] @ p2 @ [w] @ p3) v
proof (state)
this:
path G u (p1 @ [w] @ p2 @ [w] @ p2 @ [w] @ p3) v
Output
Query
Sledgehammer
Symbols
154,1 (4780/6536)
(isabelle,isabelle,UTF-8-Isabelle)
69/767MB 1:51 PM
lammich@lapnipkow10: ~/lehre/FDS/SS1...
Isabelle2017 - tut05.thy
13:51:05


Isabelle2017 - tut05.thy
File Edit Search Markers Folding View Utilities Macros Plugins Help
tut05.thy (~/lehre/FDS/SS18/public/exercises/)
shows "∃p'. length p' > length p ∧ ¬distinct p' ∧ path G u p' v"
proof -
from not_distinct_decomp[OF ND] obtain p1 p2 p3 w where
[simp]: "p = p1@[w]@p2@[w]@p3" by blast
from P obtain w' w'' where "path G u p1 w'" "G w' w" "path G w p2 w''" "G w'' w" "path G w p3 v"
by auto
hence P: "path G u (p1@[w]@p2@[w]@p2@[w]@p3) v" (is "path _ _ ?p' _") by auto
have "length ?p' > length p ∧ ¬distinct ?p'" by auto
with P show ?thesis by blast
qed
text ‹
Proof state
Auto update
Update
Search:
100%
have P: path G u (p1 @ [w] @ p2 @ [w] @ p2 @ [w] @ p3) v
proof (state)
this:
path G u (p1 @ [w] @ p2 @ [w] @ p2 @ [w] @ p3) v
Output
Query
Sledgehammer
Symbols
155,1 (4781/6536)
(isabelle,isabelle,UTF-8-Isabelle)
85/767MB 1:51 PM
lammich@lapnipkow10: ~/lehre/FDS/SS1...
Isabelle2017 - tut05.thy
13:51:22


Isabelle2017 - tut05.thy
File Edit Search Markers Folding View Utilities Macros Plugins Help
tut05.thy (~/lehre/FDS/SS18/public/exercises/)
text ‹
\NumHomework{Estimate Recursion Equation}{May 18}
(Sledgehammer allowed again)
Show that the function defined by ‹a 0 = 0› and ‹a (n+1) = (a n)² + 1›
is bounded by the double-exponential function ‹2^(2^n)›
›
fun a :: "nat ⇒ int" where
"a 0 = 0" |
"a (Suc n) = a n ^ 2 + 1"
text ‹
Proof state
Auto update
Update
Search:
100%
Output
Query
Sledgehammer
Symbols
185,1 (5664/6536)
(isabelle,isabelle,UTF-8-Isabelle)
59/767MB 1:51 PM
lammich@lapnipkow10: ~/lehre/FDS/SS1...
Isabelle2017 - tut05.thy
13:51:53


Isabelle2017 - tut05.thy
File Edit Search Markers Folding View Utilities Macros Plugins Help
tut05.thy (~/lehre/FDS/SS18/public/exercises/)
Show that the function defined by ‹a 0 = 0› and ‹a (n+1) = (a n)² + 1›
is bounded by the double-exponential function ‹2^(2^n)›
›
fun a :: "nat ⇒ int" where
"a 0 = 0" |
"a (Suc n) = a n ^ 2 + 1"
text ‹
We have given you a proof skeleton, setting up the induction.
To complete your proof, you should come up with a chain of inequations.
You may try to solve the intermediate steps with sledgehammer.
Hint: It is a bit tricky to get the approximation right.
Proof state
Auto update
Update
Search:
100%
Output
Query
Sledgehammer
Symbols
190.1 (5737/6536)
(isabelle,isabelle,UTF-8-Isabelle)
1:52 PM


Isabelle2017 - tut05.thy
File Edit Search Markers Folding View Utilities Macros Plugins Help
tut05.thy (~/lehre/FDS/SS18/public/exercises/)
We strongly recommend to sketch the inequations on paper first.
Hint: Have a look at the lemma @{thm [source] power_mono}, in particular its
instance for squares:
›
thm power_mono[where n=2]
lemma "a n ≤ 2 ^ (2 ^ n) - 1"
proof(induction n)
case 0 thus ?case by simp
next
case (Suc n)
assume IH: "a n ≤ 2 ^ 2 ^ n - 1"
Proof state
Auto update
Update
Search:
100%
proof (prove)
goal (1 subgoal):
1. a n ≤ 2 ^ 2 ^ n - 1
Output
Query
Sledgehammer
Symbols
203.29 (6230/6536)
(isabelle,isabelle,UTF-8-Isabelle)
68/760MB
1:52 PM


Isabelle2017 - tut05.thy
File Edit Search Markers Folding View Utilities Macros Plugins Help
tut05.thy (~/lehre/FDS/SS18/public/exercises/)
We strongly recommend to sketch the inequations on paper first.
Hint: Have a look at the lemma @{thm [source] power_mono}, in particular its
instance for squares:
›
thm power_mono[where n=2]
lemma "a n ≤ 2 ^ (2 ^ n) -1"
proof(induction n)
case 0 thus ?case by simp
next
case (Suc n)
assume IH: "a n ≤ 2 ^ 2 ^ n - 1"
Proof state
Auto update
Update
Search:
100%
proof (prove)
goal (1 subgoal):
1. a n ≤ 2 ^ 2 ^ n - 1
Output
Query
Sledgehammer
Symbols
203.27 (6228/6536)
(isabelle,isabelle,UTF-8-Isabelle)
70/760MB
1:53 PM