

Automated Reasoning 2018/2019

Assignment: Theorem Proving in Isabelle

Jacques Fleuriot Imogen Morris

October 16, 2018

Introduction

The practical assignment for students on the Automated Reasoning course involves theorem proving in Isabelle. You will be required to formalise a few axioms about a geometry of oriented curves and then combine these with the rules of logic to mechanically prove a number of geometric theorems.

Isabelle

Isabelle is a generic interactive theorem prover. This means that Isabelle can be used to formalise theorems in various logics. For this practical, you will be using Isabelle/HOL, which is the higher-order logic of Isabelle. To get started, download the file `practical.thy` from:

<http://www.inf.ed.ac.uk/teaching/courses/ar>

Essential Reading

As you will be using Isabelle interactively, you will need to be familiar with the system before you start. Formal mathematics is not trivial! You will find this assignment much easier if you attend the lectures, attempt the various Isabelle exercises given on the course webpages, and ask questions about using Isabelle before you start. It is recommended that you read Chapter 5 of the Isabelle/HOL tutorial located at:

<http://www.cl.cam.ac.uk/research/hvg/Isabelle/documentation.html>

Some Useful Commands

Isabelle has many commands which will help you mechanise the theorems in this practical. You should refer to the Isabelle tutorial and lectures to discover the commands available. One of the built-in methods you should be aware of is called **auto**. It uses both the *classical reasoner* and simplifier of Isabelle. The command **apply auto** tells Isabelle to apply **auto** to all subgoals. You are allowed to use this command in Parts 2 and 3 of the practical.

You should also be aware of the tactic **cut_tac**. This inserts a known rule or fact as an assumption in your proof. For example, the known fact:

excluded_middle: $\neg P \vee P$

can be inserted as an assumption in your proof by using the command

apply (cut_tac excluded_middle)

If you wish to rename the variable P , to A say, then you can simply give the command

apply (cut_tac P=A in excluded_middle)

If you are struggling to mechanise a lemma or theorem in Isabelle, then the command **sorry** can be used. This allows the lemma or theorem to be asserted as true without completing the proof. It will enable you to make progress in the practical, however no marks will be allocated for the missing part of the proof. You should not use other people's proofs or formalisations.

Part 1: Some propositional and first-order proofs [25%]

For the first part of this assignment, you should attempt to prove a number of simple propositional and first-order statements in Isabelle, and to provide a description of your experience.

For this part of the assignment use only the following proof methods: **rule**, **rule_tac**, **drule**, **drule_tac**, **erule**, **erule_tac**, **frule**, **frule_tac**, **cut_tac** and **assumption**. You are also restricted to using only the following Natural Deduction rules: **conjI**, **conjE**, **implI**, **impE**, **mp**, **iffI**, **iffE**, **notI**, **notE**, **disjI1**, **disjI2**, **disjE**, **exI**, **exE**, **allI**, **allE** and **spec**. You are also

allowed to use **excluded middle**. You may also use as rules any lemmas that you have proven in this way.

Attempt proofs of the following statements:

- $(\neg P \vee R) \longrightarrow (P \longrightarrow R)$ (1 mark)
- $(P \longrightarrow Q \longrightarrow R) \longrightarrow P \wedge Q \longrightarrow R$ (1 mark)
- $(P \longrightarrow R) \longrightarrow (\neg P \vee R)$ (2 marks)
- $(P \longrightarrow Q) \wedge (\neg P \longrightarrow R) \longrightarrow R \vee Q$ (2 marks)
- $((P \longrightarrow Q) \longrightarrow P) \longrightarrow P$ (4 marks)
- $(\neg P \vee \neg R) \longleftrightarrow (\neg(P \wedge R))$ (5 marks)
- $P a \longrightarrow (\forall x. P x \longrightarrow F x) \longrightarrow F a$ (2 mark)
- $(\forall x. F x \wedge G x) \longrightarrow ((\forall x. F x) \wedge (\forall x. G x))$ (2 mark)
- $(\forall x. P x \longrightarrow \neg F x) \longrightarrow \neg (\exists x. P x \wedge F x)$ (3 marks)
- $(\neg (\forall x. \neg P x)) \vee (\exists x. \neg P x)$ (3 marks)

Part 2: A Geometry of Simple Curves

In some work on qualitative geometry, Kulik and Eschenbach present a formal *axiomatic* framework dealing with oriented curves [1]. This work introduces points (denoted by P, Q, P', \dots), curves (denoted by c, c_1, \dots), and oriented curves (denoted by o, o_1, \dots) as primitive geometric entities (structures) and two primitive relations, namely, *incidence* and *precedence*. A number of basic definitions and axioms are then given to characterise the relationships between these various primitive entities.

In this assignment, your task is to formalise part of this axiomatic framework (which only involves simple, non-oriented curves and the incidence relation) and mechanically prove a number of theorems as given in paper [1]. Your work will thus provide a rigorous, mechanical verification of some of the claims made by Kulik and Eschenbach.

Note that for this part of the assignment your proof should **not** use the methods **smt**, **metis**, **meson**, **presburger** or **moura**.

Mechanizing the Basic Definitions of Simple Curves [15%]

The binary incidence relation is given in the locale

```
locale incidence =
  fixes incident ::
    "[[pt, 'curve]] ⇒ bool" ("_ isIncidentTo _" [60, 60] 60)
```

The predicate takes two arguments and represent the notion of a point lying on a curve. It has been declared as an *infix* predicate, so you can express that a point P lies on the curve c by **P isIncidentTo c**. In the template file `practical.thy`, you have been provided with the declared, but not yet defined, predicates **isPartOf** (representing the \sqsubset given by Kulik and Eschenbach [1]), **isEndPoint**, **isInnerPoint**, **MeetAt**, **isSumOf**, **isClosedCurve** and **isOpenCurve**. These are slightly more readable names – using infix notations whenever possible – for the predicates used by Kulik and Eschenbach (e.g. we can specify that a point P is an inner point of a curve c as **P isInnerPoint c** instead of **ipt (P,c)**).

Note that the *sum* operation is defined using an $=$ symbol by Kulik and Eschenbach. Although this is a convenient notational device for the paper, this is not advisable in our formalization as (among many other issues) $=$ is already defined in Isabelle. So, we explicitly define a new relation **isSumOf** such that c is the sum of c_1 and c_2 (i.e. $c_1 \sqcup c_2$) is denoted by **c isSumOf c1 c2**. Note also that one often has to make such representational choices (e.g. relational vs. functional) when dealing with the mechanization of a (pen-and-paper) framework.

Your tasks are to:

1. Formalise Definitions (2.1)-(2.4) from Kulik and Eschenbach's paper [1] in Isabelle. (7 marks)
2. Formalise and prove the remark attached to Definition 2.2, namely that:

A logically equivalent variant uses the definition of an inner point as basic notion and defines the endpoint on this basis. A point is an inner point of a curve if there are two sub-curves which include the point and none of the sub-curves is part of the other.

Make sure your proofs are structured Isar proofs of more than one line so that the proofs form explanations for why the theorems are true. Your alternative definitions for inner point and endpoint should be given as lemmas **isInnerPoint_def2** and **isEndPoint_def2** respectively in your theory file. (8 marks)

Mechanising the Axioms for Simple Curve Geometry [10%]

In your theory file you are required to formalise the curve axioms (C2)-(C9) given in Section 2.2. of Kulik and Eschenbach's paper. You are to formalise the axioms in a locale **curve_axioms** which imports the **incidence** locale. The structure of the locale has already been given, so you only need to add the axioms. Axiom (C1) has been provided as an example for you:

```
axiom_c1: "[[c' isPartOf c;c' ≠ c]] ⇒ isOpenCurve c"
```

Note that the formulation of Axiom (C1) takes advantage of the meta-level syntax available in Isabelle. The explicit universal (\forall) quantifiers present at the outer-level in Kulik and Eschenbach's pen-and-paper axioms can be omitted (turning the associated universal variables into schematic Isabelle ones that can then be instantiated - try the command **thm axiom_c1** to examine the Isabelle representation more closely). This representation makes it easier to use the axioms as Natural Deduction rules in proofs and you are strongly encouraged to follow a similar approach.

Mechanising the Basic Consequences of the Axioms [30%]

Using your formalised definitions and axioms, you will now mechanise (in your theory file) a number of resulting properties. Most of the results are taken from the early part of Section 2.3 of Kulik and Eschenbach's paper. You may be required (or find it helpful) to prove additional lemmas, not explicitly mentioned and/or named in the paper, to facilitate your mechanisation task. Express your lemmas in the style **assumes ... shows** (see for example the transitive property of Theorem 2.5 formalised below). Note that, with the exception of **smt**, **metis**, **meson**, **presburger** and **moura**, you can use Isabelle automatic tools (such as **simp**, **auto**, **blast**) in your proofs. You are expected to give readable, structured Isar proofs. It is acceptable to give one-line proofs of theorems (e.g. **by auto**) unless otherwise indicated.

You should also not use the automatically generated Isar proofs. The results that you need to represent and mechanise are:

1. Theorem 2.5: *The relation part-of (sub-curve) is an order relation i.e. that it is reflexive, transitive, and anti-symmetric.* Note that you should prove these as 3 separate results (making them easy to use later on, if need be). For instance, the transitive property could be stated as follows:

```
theorem isPartOf_trans:
  assumes "c1 isPartOf c2" "c2 isPartOf c3"
  shows "c1 isPartOf c3"
```

(2 marks)

2. Corollary 2.6: *The sum is monotone.* Note that use of relational form for sum of curves (i.e. the **isSumOf** predicate) makes the mechanised representations slightly different from the version given in the paper. The first part of the corollary (also given in file `practical.thy`) may be represented as follows:

```
lemma corollary_2_6_part1:
  assumes "c2 isPartOf c3" "c isSumOf c1 c2"
          "c' isSumOf c1 c3"
  shows "c isPartOf c'"
```

Notice that the assumptions that the curves meet are no longer needed.
(2 marks)

3. Theorem 2.7: *Every open curve has at least two endpoints, and every sub-curve of an open curve is open.* (3 marks)

4. Remark 2.8: *Every open curve has exactly two endpoints.* This statement can be represented in Isabelle as follows:

lemma remark2_8:

assumes "isOpenCurve c"

shows "card {p. p isEndPoint c} = 2"

where **card** denotes the set cardinality function in Isabelle. Using the cardinality operator, statements such as **card {x, y} = 2** can easily be proved in Isabelle. You can find useful theorems about **card** by searching for **card** in the ‘query’ box in Isabelle, or by searching the imported Isabelle theories directly (they can be found online at <https://isabelle.in.tum.de/library/HOL/>). Write a structured Isar proof of more than one line so that the proof forms an explanation for why the theorem is true. (5 marks)

5. Corollary 2.9: *If one curve is part of another curve then they cannot meet.* Formalise the proof that Kulik and Eschenbach give as a structured Isar proof. Your proof should reflect the reasoning used in the pen-and-paper proof. (4 marks)
6. Theorem 2.10: *If an endpoint of a given curve lies on a sub-curve then it is also an endpoint of this sub-curve.* Formalise the proof that Kulik and Eschenbach give as a structured Isar proof. Your proof should reflect the reasoning used in the pen-and-paper proof. (5 marks)
7. Corollary 2.11: *Inner points of a sub-curve of any curve c are inner points of c.* Write a structured Isar proof of more than one line so that the proof forms an explanation for why the theorem is true. (4 marks)
8. Corollary 2.12: *If P is a meeting point of two curves and lies on a sub-curve of one of the two curves then P is also a meeting point of the sub-curve and the other curve.* Write a structured Isar proof of more than one line so that the proof forms an explanation for why the theorem is true. (5 marks)

Part 3: Challenge Proof: Theorem 2.13 [20%]

In this part, you should:

1. Attempt a mechanical proof of Theorem 2.13, which states that:

Two distinct points on an open curve uniquely determine the sub-curve connecting these points.

You must again give readable, structured Isar proofs. You may not use **smt**, **presburger** or **moura** but you may use **metis** and **meson**, e.g. via the invocation of sledgehammer. Note that this part of the assignment is challenging and may involve a lot of effort without a proper mechanisation plan. You are strongly advised to follow the proof outlined in the paper. In particular, you should consider Step 1 and Step 2 separately. You may find it helpful to prove various sub-lemmas. (16 marks)

Note also that credit will be given for an unsuccessful or incomplete mechanisation attempt that proved *significant* lemmas/theorems demonstrating progress towards a final proof.

2. In under 300 words, compare and contrast the mechanical proof (or part(s) of proof) that you produced with the pen-and-paper proof by Kulik and Eschenbach. In particular, indicate any reasoning arguments, proof parts, and/or useful lemmas that you had to make explicit during the mechanisation but may have been glossed over or assumed by the pen-and-paper proof. (4 marks)

Demonstrator Hours and Help

The demonstrator, Imogen Morris (s1402592@ed.ac.uk), will be available to give advice on Mondays, 9am-11am in 5.05 - West Lab, Appleton Tower.

You are strongly encouraged to make use of the Piazza forum for discussion of general problems and for sharing any queries that you may have.

Note that, although we encourage discussions about the assignment, you must not discuss or share actual proof scripts (i.e. solutions) to any of the problems with fellow students.

Submission

By 4pm on 19th November 2018 you must submit your solution in electronic form. This should consist of your theory file **practical .thy** and can be submitted using the command:

```
submit ar cw1 practical .thy
```

Late coursework will be penalised in accordance with the Informatics standard policy (see <http://edin.ac/1LRb1YG>). Please consult your course guide for specific information about this. Also note that, while we encourage students to discuss the practical among themselves, we take plagiarism **seriously** and any suspected case will be treated appropriately. Please remember the University requirements as regards all assessed work. Details about this can be found at:

<http://web.inf.ed.ac.uk/infweb/admin/policies/academic-misconduct>

Furthermore, you are required to take reasonable measures to protect your assessed work from unauthorised access. For example, if you put any such work on a public repository then you must set access permissions appropriately (generally permitting access only to yourself).

References

- [1] L. Kulik and C. Eschenbach. A Geometry of Oriented Curves. Report from the project: Axiomatics of Spatial Concepts. Department for Informatics, University of Hamburg, 1999. Available at:

<http://www.inf.ed.ac.uk/teaching/courses/ar/TROrientedCurves.pdf>.