Proof Reconstruction

Diego Alejandro Montoya-Zapata

EAFIT University

February, 2015

Some Definitions

- **Agda** is a proof assistant. It is an interactive system for writing and checking proofs. Agda is also a functional language with dependent types.

[1] Bove and Dybjer (2009), “Dependent types at work”

[2] `http://www.cs.miami.edu/~tptp/OverviewOfATP.html`

Some Definitions

- **Agda** is a proof assistant. It is an interactive system for writing and checking proofs. Agda is also a functional language with dependent types.
- "A **dependent type** is a type that depends on elements of other types. An example is the type A^n of vectors of size n with components of type A."[1]

[1] Bove and Dybjer (2009), "Dependent types at work"

[2] `http://www.cs.miami.edu/~tptp/OverviewOfATP.html`

Some Definitions

- **Agda** is a proof assistant. It is an interactive system for writing and checking proofs. Agda is also a functional language with dependent types.
- "A **dependent type** is a type that depends on elements of other types. An example is the type A^n of vectors of size n with components of type A."[1]
- "**Automated Theorem Proving** (ATP) deals with the development of computer programs that show that some statement (*conjecture*) is a logical consequence of a set of statements (the *axioms* and *hypotheses*)."[2]

[1] Bove and Dybjer (2009), "Dependent types at work"

[2] `http://www.cs.miami.edu/~tptp/OverviewOfATP.html`

Some Definitions

- **Agda** is a proof assistant. It is an interactive system for writing and checking proofs. Agda is also a functional language with dependent types.
- “A **dependent type** is a type that depends on elements of other types. An example is the type A^n of vectors of size n with components of type A.”[1]
- “**Automated Theorem Proving** (ATP) deals with the development of computer programs that show that some statement (*conjecture*) is a logical consequence of a set of statements (the *axioms* and *hypotheses*).”[2]
- **TPTP** is a language understood by most of the ATPs.

[1] Bove and Dybjer (2009), “Dependent types at work”

[2] `http://www.cs.miami.edu/~tptp/OverviewOfATP.html`

Some Definitions

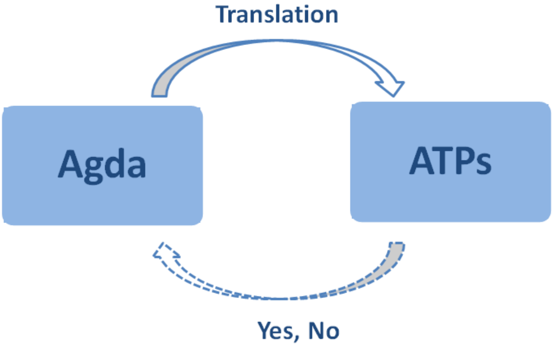
- **Agda** is a proof assistant. It is an interactive system for writing and checking proofs. Agda is also a functional language with dependent types.
- "A **dependent type** is a type that depends on elements of other types. An example is the type A^n of vectors of size n with components of type A."[1]
- "**Automated Theorem Proving** (ATP) deals with the development of computer programs that show that some statement (*conjecture*) is a logical consequence of a set of statements (the *axioms* and *hypotheses*)."[2]
- **TPTP** is a language understood by most of the ATPs.
- **Apia** is a program (developed by Professor Andrés Sicard-Ramírez) that performs the translation of an Agda representation of FOL formula into TPTP.

[1] Bove and Dybjer (2009), "Dependent types at work"

[2] `http://www.cs.miami.edu/~tptp/OverviewOfATP.html`

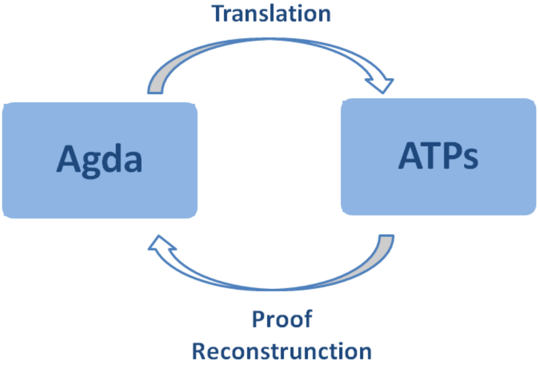
Problem Definition

In this moment:



Problem Definition

What we want:



Goal

The TPTP library has provided the community with standards for input and output for ATPs [3]. However, it does not exist a standard for the way the proof is printed, which make it difficult to try to do a program to reconstruct the proofs for all of the ATPs. For this reason, we decided to focus our efforts in formulating the demonstration in Agda just for one ATP.

In addition, some of the ATPs are not developed for Windows. Then, the ATP chosen to perform the reconstruction of its proofs was **SPASS**.

[3] `http://www.cs.miami.edu/~tptp/`

State of the Art

- **SMTCoq** is a Coq tool that checks proof witnesses coming from external SAT and SMT solvers. [4]

[4] Armand, Faure, Grégoire, Keller, Théry and Werner (2011), “A Modular Integration of SAT/SMT Solvers to Coq through Proof Witnesses”

[5] Blanchette, Bohme, Fleury, Smolka and Steckermeier (2015), “Semi-intelligible Isar Proofs from Machine Generated Proofs”

[6] Foster and Struth (2011), “Integrating an Automated Theorem Prover into Agda”

State of the Art

- **SMTCoq** is a Coq tool that checks proof witnesses coming from external SAT and SMT solvers. [4]
- **Sledgehammer** is a component of the Isabelle/HOL proof assistant that integrates external ATPs to discharge interactive proof obligations. Something impressive is that Sledgehammer transforms the proofs by contradiction into direct proofs. [5]

[4] Armand, Faure, Grégoire, Keller, Théry and Werner (2011), “A Modular Integration of SAT/SMT Solvers to Coq through Proof Witnesses”

[5] Blanchette, Bohme, Fleury, Smolka and Steckermeier (2015), “Semi-intelligible Isar Proofs from Machine Generated Proofs”

[6] Foster and Struth (2011), “Integrating an Automated Theorem Prover into Agda”

State of the Art

- **SMTCoq** is a Coq tool that checks proof witnesses coming from external SAT and SMT solvers. [4]
- **Sledgehammer** is a component of the Isabelle/HOL proof assistant that integrates external ATPs to discharge interactive proof obligations. Something impressive is that Sledgehammer transforms the proofs by contradiction into direct proofs. [5]
- Foster and Struth integrated the Waldmeister ATP to Agda. [6]

[4] Armand, Faure, Grégoire, Keller, Théry and Werner (2011), "A Modular Integration of SAT/SMT Solvers to Coq through Proof Witnesses"

[5] Blanchette, Bohme, Fleury, Smolka and Steckermeier (2015), "Semi-intelligible Isar Proofs from Machine Generated Proofs"

[6] Foster and Struth (2011), "Integrating an Automated Theorem Prover into Agda"