Focused search for arguments from propositional knowledge

Vasiliki Efstathiou and Anthony Hunter

Department of Computer Science
University College London
Contents

▶ Introduction: Framework for argumentation and motivation for efficient search for arguments
▶ The connection graph approach, definitions and algorithm demonstration
▶ Theoretical and experimental results
▶ Conclusions and further work
Framework for argumentation (Besnard & Hunter 2001)

- We can formalize argumentation using classical logic and adapt it in computational context.
- We use $\Delta, \Phi, \ldots$ to denote sets of formulae, $\phi, \psi \ldots$ to denote formulae and $a, b, c \ldots$ to denote the propositional letters each formula consists of.
- In this framework an argument is a pair $\langle \Psi, \phi \rangle$ where $\Psi$ is a set of formulae that minimally and consistently entails a formula $\phi$. We call $\Psi$ the **support** of the argument and $\phi$ the **claim** of the argument.
Examples

Some arguments are

- $\langle \{\neg a, (d \lor e) \land f\}, \neg a \land (d \lor e) \rangle$
- $\langle \{(\neg a \lor b) \land c, \neg b \land d\}, \neg a \land c \rangle$
- $\langle \{\neg a\}, \neg a \rangle$
- $\langle \{\neg b \land d\}, d \rangle$
Motivation for efficient algorithms

- We want to automate the construction of arguments.
- This process is computationally expensive.
- Given a knowledgebase $\Delta$, we want to find all the arguments for a formula $\phi$.
- We use an automated theorem prover (ATP) to test for entailment and consistency
  - $\psi \vdash \phi$?
  - $\psi \not\vdash \bot$?
Motivation for efficient algorithms

- We do not know which subsets of $\Delta$ to investigate. Testing arbitrary subsets of $\Delta$ can be prohibitely expensive. We explore an alternative way for locating the arguments for $\phi$.

- Our approach is to adapt the idea of connection graphs (R. Kowalski 1975) to reduce the search space for argumentation.

- We use this in order to isolate a partition of the knowledgebase that contains the arguments for $\phi$. 
Definitions

We start with a language of disjunctive clauses (disjunctions of 1 or more literals). We define the following relations on clauses:

- The Disjuncts relation takes a clause and returns the set of disjuncts in the clause. \( \text{Disjuncts}(\beta_1 \lor \ldots \lor \beta_n) = \{\beta_1, \ldots, \beta_n\} \)

- Let \( \phi \) and \( \psi \) be clauses. Then, \( \text{Preattacks}(\phi, \psi) \) is \( \{\beta \mid \beta \in \text{Disjuncts}(\phi) \text{ and } \neg \beta \in \text{Disjuncts}(\psi)\} \)

- Let \( \phi \) and \( \psi \) be clauses. If \( \text{Preattacks}(\phi, \psi) = \{\beta\} \) for some \( \beta \), then \( \text{Attacks}(\phi, \psi) = \beta \) otherwise \( \text{Attacks}(\phi, \psi) = \text{null} \)
Examples

▶ Preattacks

▶ Preattacks($a \lor \neg b \lor \neg c \lor d$, $a \lor b \lor \neg d \lor e$) = $\{\neg b, d\}$
▶ Preattacks($a \lor b \lor \neg d \lor e$, $a \lor \neg b \lor \neg c \lor d$) = $\{b, \neg d\}$
▶ Preattacks($a \lor b \lor \neg d$, $a \lor b \lor c$) = $\emptyset$
▶ Preattacks($a \lor b \lor \neg d$, $a \lor b \lor d$) = $\{-d\}$
▶ Preattacks($a \lor b \lor \neg d$, $e \lor c \lor d$) = $\{-d\}$

▶ Attacks

▶ Attacks($a \lor \neg b \lor \neg c \lor d$, $a \lor b \lor \neg d \lor e$) = $null$
▶ Attacks($a \lor b \lor \neg d \lor e$, $a \lor \neg b \lor \neg c \lor d$) = $null$
▶ Attacks($a \lor b \lor \neg d$, $a \lor b \lor c$) = $null$
▶ Attacks($a \lor b \lor \neg d$, $a \lor b \lor d$) = $\neg d$
▶ Attacks($a \lor b \lor \neg d$, $e \lor c \lor d$) = $\neg d$
Connection graphs

- We use Preattacks and Attacks relations on a set of clauses $\Delta$ to define different types of graphs
- The nodes of the graphs are elements from $\Delta$
- Arcs exists between nodes which contain contradictory literals
- The number of contradictory literals between pairs of nodes allows for different relations to hold between those nodes, which in turn identify different kinds of graphs
The connection Graph

The connection graph is the graph whose arcs are identified by the Pre-attacks relation:

\[
\begin{align*}
\neg b & \quad \neg c \lor \neg g & \quad \neg c & \quad \neg h \lor l & \quad \neg l \lor \neg k & \quad n \lor m \lor \neg q \\
a \lor b & \quad \neg b \lor d & \quad c \lor g & \quad h \lor \neg l & \quad l \lor k & \quad \neg n \quad \neg m \quad q \\
\neg a \lor d & \quad \neg d & \quad \neg g & \quad f \lor p & \quad \neg k & \quad m
\end{align*}
\]
The **attack graph** is the graph whose arcs are identified by the Attacks relation

\[
\begin{align*}
\neg b & \quad \neg c \lor \neg g & \quad \neg c & \quad \neg h \lor l & \quad \neg l \lor \neg k & \quad n \lor m \lor \neg q \\
\quad & \quad \quad & \quad & \quad & \quad & \\
\quad a \lor b & \quad \neg b \lor d & \quad c \lor g & \quad h \lor \neg l & \quad l \lor k & \quad \neg n & \quad \neg m & \quad q \\
\quad & \quad \quad & \quad & \quad & \quad & \\
\neg a \lor d & \quad \neg d & \quad \neg g & \quad f \lor p & \quad \neg k & \quad m
\end{align*}
\]
The closed graph

The closed graph characterizes the attack graph in terms of connectivity. Clauses containing ‘unlinked literals’ are excluded.

\[\neg b \quad \neg c \quad n \lor m \lor \neg q\]
\[\begin{align*}
a \lor b & \quad \neg b \lor d & \quad c \lor g & \\
\neg a \lor d & \quad \neg d & \quad \neg g & \\
\neg n & \quad \neg m & \quad q & \quad m
\end{align*}\]
The focal graph

- The **focal graph** is identified by a clause $\phi$ from $\Delta$, which we call the **epicentre**. The focal graph of $\phi$ in $\Delta$ is the component of the closed graph that contains $\phi$.

- The following is the focal graph of $\neg b$ in $\Delta$ and of $a \lor b$ in $\Delta$ and of $\neg b \lor d$ in $\Delta$ etc...

\[
\begin{align*}
\neg b \\
| \\
\begin{aligned}
  a \lor b & \quad \neg b \lor d \\
  | & \quad | \\
  \neg a \lor d & \quad \neg d
\end{aligned}
\end{align*}
\]
Algorithm for the focal graph

- Given a clause $\phi$ we can find the focal graph of $\phi$ in $\Delta$ by depth-first search of the attack graph for $\Delta$
- The following is the attack graph for a set of clauses $\Delta$. We want to find the focal graph of $\neg c$ in $\Delta$

$\neg c \rightarrow \neg b \lor c \lor d \rightarrow b \lor \neg p$

$\neg d \lor m \quad \neg d \lor p$

$\neg m \lor n$

$b \lor \neg c \lor k \rightarrow \neg k \lor e$

$\neg e \lor f \lor g$

$\neg f \quad \neg g$
Algorithm for the focal graph

- Initially all the nodes are considered to be allowed candidates for the focal graph and the unsuitable ones will be rejected while walking over the graph
- First locate $\neg c$ in the attack graph for $\Delta$

\[
\begin{align*}
\neg c &\quad \neg b \lor c \lor d \quad b \lor \neg p \\
&\quad \neg d \lor m \quad \neg d \lor p \\
&\quad \neg m \lor n \\
\end{align*}
\]

\[
\begin{align*}
b \lor \neg c \lor k &\quad \neg k \lor e \\
&\quad \neg e \lor f \lor g \\
\end{align*}
\]

$\neg f \quad \neg g$

Vasiliki Efstathiou and Anthony Hunter

Focused search for arguments from propositional knowledge
Algorithm for the focal graph

- Follow one of the paths that start from $\neg c$

$\neg c \rightarrow \neg b \lor c \lor d \rightarrow b \lor \neg p$

$\neg d \lor m \quad \neg d \lor p$

$\neg m \lor n$

$b \lor \neg c \lor k \rightarrow \neg k \lor e$

$\neg e \lor f \lor g$

$\neg f \quad \neg g$
Algorithm for the focal graph

- follow one of the paths that start from \( \neg c \)
- test if the current node is connected i.e. if all its disjuncts correspond to a link in the graph

\[
\begin{align*}
\neg c &\quad \neg b \lor c \lor d \quad \rightarrow \quad b \lor \neg p \\
\quad \downarrow &\quad \downarrow &\quad \downarrow \\
\neg d \lor m &\quad \neg d \lor p \\
\quad \downarrow &\quad \downarrow \\
\neg m \lor n &\quad \neg e \lor f \lor g \\
\neg f &\quad \neg g
\end{align*}
\]
Algorithm for the focal graph

- if it is, follow one of the paths that continue from this node

\[-c \rightarrow \neg b \lor c \lor d \rightarrow b \lor \neg p \]

\[-d \lor m \rightarrow \neg m \lor n\]

\[b \lor \neg c \lor k \rightarrow \neg k \lor e\]

\[-e \lor f \lor g\]

\[-f \quad \neg g\]
Algorithm for the focal graph

- If it is, follow one of the paths that continue from this node
- Test if the current node is connected

\[
\begin{align*}
\neg c \rightarrow \neg b \vee c \vee d \rightarrow b \vee \neg p & \quad \text{and} \quad b \vee \neg c \vee k \rightarrow \neg k \vee e \\
\neg d \vee m & \quad \neg d \vee p & \quad \neg e \vee f \vee g \\
\neg m \vee n & \quad \neg f & \quad \neg g
\end{align*}
\]
Algorithm for the focal graph

- if it is, follow one of the paths that continue from this node
- continue in the same way for every newly created node

\[
\neg c \rightarrow \neg b \lor c \lor d \rightarrow b \lor \neg p
\]

\[
\neg d \lor m \quad \neg d \lor p
\]

\[
\neg m \lor n
\]

\[
b \lor \neg c \lor k \rightarrow \neg k \lor e
\]

\[
\neg e \lor f \lor g
\]

\[
\neg f \quad \neg g
\]
Algorithm for the focal graph

- if a node which is not connected is found then mark it as rejected and backtrack

\[
\neg c \rightarrow \neg b \lor c \lor d \rightarrow b \lor \neg p \\
\neg d \lor m \rightarrow \neg d \lor p \\
\neg m \lor n
\]

\[
b \lor \neg c \lor k \rightarrow \neg k \lor e \\
\neg e \lor f \lor g \\
\neg f \lor \neg g
\]
Algorithm for the focal graph

- if a node which is not connected is found then mark it as rejected and backtrack

\[
\sim c \rightarrow \sim b \lor c \lor d \\
\sim d \lor m \\
\sim m \lor n
\]

\[
b \lor \sim c \lor k \rightarrow \sim k \lor e \\
\sim e \lor f \lor g \\
\sim f \sim g
\]
Algorithm for the focal graph

- test if the nodes adjacent to the node rejected last remain connected

\[
\begin{align*}
\neg c &\rightarrow \neg b \lor c \lor d \rightarrow b \lor \neg p \\
\neg d \lor m &\rightarrow \neg d \lor p \\
\neg m \lor n &
\end{align*}
\]

\[
\begin{align*}
b \lor \neg c \lor k &\rightarrow \neg k \lor e \\
\neg e \lor f \lor g &
\end{align*}
\]

\[
\begin{align*}
\neg f & \quad \neg g
\end{align*}
\]
Algorithm for the focal graph

- test if the nodes adjacent to the node rejected last remain connected
- if they do not, mark them as rejected and continue backtracking
Algorithm for the focal graph

- test if the nodes adjacent to the node rejected last remain connected

\[
\neg c \lor \neg b \lor c \lor d \lor b \lor \neg p \\
\neg d \lor m \\
\neg m \lor n \\
\]

\[
b \lor \neg c \lor k \lor k \lor e \\
\neg e \lor f \lor g \\
\neg f \lor \neg g
\]
Algorithm for the focal graph

- Test if the nodes adjacent to the node rejected last remain connected
- If they do, continue from that point, by following one of the paths to the nodes that have not been visited yet

\[
\neg c \rightarrow \neg b \lor c \lor d \rightarrow b \lor \neg p
\]

\[
b \lor \neg c \lor k \rightarrow \neg k \lor e
\]

\[
\neg d \lor m \quad \neg d \lor p
\]

\[
\neg m \lor n
\]

\[
\neg e \lor f \lor g
\]

\[
\neg f \quad \neg g
\]
Algorithm for the focal graph

- and continue in the same way. Only the component of the graph that is linked to $\neg c$ is being searched.
- The visited non-rejected nodes of the graph correspond to the focal graph of $\neg c$ in $\Delta$.

\[ \neg c \rightarrow \neg b \lor c \lor d \rightarrow b \lor \neg p \]
\[ \begin{align*}
\neg d \lor m & \hspace{1cm} \neg d \lor p \\
\neg m \lor n & 
\end{align*} \]
\[ b \lor \neg c \lor k \rightarrow \neg k \lor e \]
\[ \begin{align*}
\neg e \lor f \lor g & \\
\neg f & \neg g 
\end{align*} \]
Why is the focal graph useful?

- The focal graph can be used to reduce the search space for argumentation for knowledgebases and queries in CNF.

- Let $\text{Conjuncts}(\phi)$ be the set of clauses a formula $\phi$ in CNF consists of.

- Let $\text{SetConjuncts}(\psi)$ be the set of clauses all the formulae from $\psi$ consist of.
Theoretical results

Why is the focal graph useful?

- Let $\phi$ be a claim for which we want to find arguments from $\Psi$, where $\Psi$ be a set of formulae in CNF
- Let $\bar{\phi} = \phi_1 \land \ldots \land \phi_n$ be the CNF of the negation of claim $\phi$
- The focal graphs of each $\phi_i$ in $\text{SetConjuncts}(\Psi \cup \{\bar{\phi}\})$ indicate the part of $\Psi$ which contains the arguments for $\phi$ and hence help excluding some other which is not relevant
- We call the graph consisting of these focal graphs the query graph of $\phi$ in $\Psi$
The query graph

Let $\Psi$ be set of formulae in CNF

$$\Psi = \{ (\neg a \lor d) \land (\neg c \lor \neg g), \neg d, \neg d \land (\neg h \lor l), q \land (\neg h \lor l),$$

$$c \lor g, \neg g, \neg b, \neg b \lor d, l \lor k, m \land (\neg l \lor \neg k),$$

$$\neg k \land (n \lor m \lor \neg q), (h \lor \neg l), \neg m \land \neg n, m \land q \}$$

Let $\phi$ be a claim for an argument with $\phi = (a \lor b) \land (f \lor p) \land \neg c$
The query graph

Then, \( \text{SetConjuncts}(\Psi \cup \{\overline{\phi}\}) \) is \( \Delta \) from the first example with the following attack graph where the conjuncts of \( \overline{\phi} = (a \lor b) \land (f \lor p) \land \neg c \) are marked.

\[
\begin{array}{cccccccc}
\neg b & \neg c \lor \neg g & \neg c & \neg h \lor l & \neg l \lor \neg k & n \lor m \lor \neg q \\
| & | & | & | & | \\
| a \lor b | \neg b \lor d & c \lor g & h \lor \neg l | l \lor k | | | \\
| | | | | | | | \\
| | | | | | | | \\
| | | | | | | | \\
| | | | | | | | \\
| | | | | | | | \\
| | | | | | | | \\

\end{array}
\]
The query graph

- and so the following is the query graph of $\phi$ in $\Psi$
- We want to find arguments for $\phi$ from $\Psi$ and not from $\text{SetConjuncts}(\Psi \cup \{\overline{\psi}\})$

\[
\begin{align*}
\neg b & \quad \neg c \\
\quad a \lor b & \quad \neg b \lor d \\
\quad c \lor g & \\
\quad \neg a \lor d & \quad \neg d \\
& \quad \neg g
\end{align*}
\]
The query graph

The query graph indicates which subsets of $\Psi$ are useful - find which formula from $\Psi$ each node relates to

$$\Psi = \{ (\neg a \lor d) \land (\neg c \lor \neg g), \neg d, \neg d \land (\neg h \lor l), q \land (\neg h \lor l), c \lor g, \neg g, \neg b, \neg b \lor d, l \lor k, m \land (\neg l \lor \neg k), \neg k \land (n \lor m \lor \neg q), (h \lor \neg l), \neg m \land \neg n, m \land q \}$$
Supportbase

- Use this part of the knowledgebase to look for arguments instead of searching the initial knowledgebase

$$\Psi' = \{(\neg a \lor d) \land (\neg c \lor \neg g), \neg d, \neg d \land (\neg h \lor l), c \lor g, \neg g, \neg b, \neg b \lor d\}$$

- We call $\Psi'$ the Supportbase for $\Psi$ and $\phi$. If $\langle \Gamma, \phi \rangle$ is an argument then $\Gamma$ is a subset of the Supportbase

- $\text{Supportbase}(\Psi, \phi) \subseteq \Psi$
Experimental results

Experiment

- We tested the focal graph algorithm for sets of randomly generated clauses.
- These sets were of fixed cardinality (600 clauses) and they contained 3-place clauses (rules) and 1-place clauses (facts).
- The evaluation was based on the size of the focal graph of an epicentre $\phi$ in a set of clauses $\Delta$. 
Experimental results

Experiment

- 2 dimensions were considered:
  - clauses-to-variables ratio
  - facts-to-rules ratio

- e.g. knowledgebase with 600 elements:
  - 150 facts + 450 rules, facts-to-rules = 1/3
  - constructed with 100 propositional letters:
    clauses-to-variables ratio = 6 = 600/100

- 1000 repetitions of the algorithm for each fixed clauses-to-variables and facts-to-rules ratio

- Highest average focal graph size of an epicentre $\phi$ in a set of clauses $\Delta$ with 600 distinct elements is $\sim 344$
  (57 % of the initial knowledgebase)
Experimental data

Figure: Focal graph size variation with the clauses-to-variables ratio
Conclusions

- In this talk we presented the theoretical background of algorithms that can make argumentation more effective in terms of computational cost by reducing the search space for arguments.

- We presented some empirical results on how this proposal works with random data.

- Further work in this framework involves:
  - Algorithms for finding arguments with literals for claims and sets of clauses for supports (FOIKS ’08)
  - Generalization to subsets of first order logic
  - Experimentation with knowledgebases of real data.