# wfomi

***Release 0.1***

**korecki**

**May 26, 2020**

# CONTENTS:

# CHAPTER ONE

# PARSER

**class** parser.**Parser**

the parser for the default circuit and weight files defined by the author. The files descriptions can be found in the docs

**adjustConstNodes**(*constCorrection*)
adjusts the circuit by moving the constant nodes down when they are above a univesal quantifier over the same domain

**ancestorIsForAll**(*node*)
a helper function used in adjustConstNodes to check if the node has a universal quantifier as an ancestor

**connectNodes**()
connects the nodes in self.nodes dictionary based on data in self.connections

**nextMatchingForAll**(*node*, *domain*)
a helper function used in adjustConstNodes to detect the next universal quantifier of a given node with matching domain

**parseCircuit**(*name*, *weights*, *domains*)
parses a circuit file with the given name and creates nodes using data on the weight functions and domains

**parseConnections**(*content*)
parses the link lines of a circuit file with the given name and stores the connections between nodes in self.connections, and self.reverseConnections

**parseConst**(*line*, *domains*, *weights*, *constCorrection*, *node*)
parses a line contianing a constant node

**parseNodes**(*content*, *constCorrection*, *weights*, *domains*)
parses the node lines of a circuit file with the given name, creates and stores the nodes in the nodes dictionary

**parseQuantifier**(*line*, *domains*)
parses a line contianing a universal or existential quantifier

**parseWeights**(*name*)
parses the weight file

CHAPTER

TWO

# TERM

**class** term.**Term**(*weights=None*, *bounds=None*, *const=None*)

The Term object represent the smallest computational unit over the circuit representation. The Term is used to store the weight functions in symbolic form, the associated bounds and the constant multiplier. The term implements multiplication and addition as well as integration. The weights, bounds and constants are all lists and their elements corresponding to elements of a sum.

**integrate**()

Implements efficient integation of a term.

term.**integrateFromDict**(*wf*, *bounds*)

helper function for the integration

term.**symbolicToNumeric**(*wf*, *bounds*)

helper function for the integration

CHAPTER

THREE

# CIRCUIT

each node has a compute class which follows the computation step of the algorithm for the given node the maxDomainSize is used to compute the maximum domain size for the existential node .. moduleauthor:: Marcin Korecki

**class** circuit.**AndNode**(*left=None*, *right=None*)

**compute**(*domSize=None*, *removed=None*)
computes the symbolic value at the and node by multiplying two terms at its child nodes. the domSize and removed are passed for potential existential and universals that may be the node's descendants

**maxDomainSize**()
used to compute the maximum domain size for the existential node

**class** circuit.**ConstNode**(*data=None*, *nodeName=None*, *varList=None*, *domData=None*)

**compute**(*domSize=None*, *removed=None*)
Computes the symbolic value at the constant node depending on its type

**maxDomainSize**()
used to compute the maximum domain size for the existential node

**class** circuit.**ExistsNode**(*var=None*, *domData=None*)

**compute**(*domSize=None*, *removed=None*)
computes the symbolic value at the existential node based on the size of the domain it quantifies over and taking into account the objects removed from it

**maxDomainSize**()
used to compute the maximum domain size for the existential node

**class** circuit.**ForAllNode**(*var=None*, *domData=None*)

**compute**(*domSize=None*, *removed=None*)
computes the numerical value at the universal node based on the size of the domain it quantifies over taking into account the objects that have been removed from it

**maxDomainSize**()
used to compute the maximum domain size for the existential node

**class** circuit.**LeafNode**(*data=None*, *weights=None*)

**compute**(*domSize=None*, *removed=None*)
computes the symbolic value at the leaves depending on weather the corresponding weight is a float or a function

**maxDomainSize**()
used to compute the maximum domain size for the existential node

**class** circuit.**Node**(*left=None*, *right=None*)
The base class defining a node, all other nodes inherit from it

**maxDomainSize**()
used to compute the maximum domain size for the existential node

**class** circuit.**OrNode**(*left=None*, *right=None*)

**compute**(*domSize=None*, *removed=None*)
computes the symbolic value at the or node by adding two terms at its child nodes, the setsize and removed are passed for potential existential and universals that may be the node's descendants

**maxDomainSize**()
used to compute the maximum domain size for the existential node

# PYTHON MODULE INDEX

# INDEX