Carleton
UNIVERSITY

Stock Trading Strategy Creation Using GP on GPU

Dave McKenney

Presentation Outline

Carleton UNIVERSITY

- Stock Market and Technical Analysis Introduction
- Description of Genetic Programming Method
- Why/How to Perform GP on GPU
- Running Time Comparisons
- Profitability Comparisons
- Conclusions and Future Work

The Stock Market

Carleton UNIVERSITY

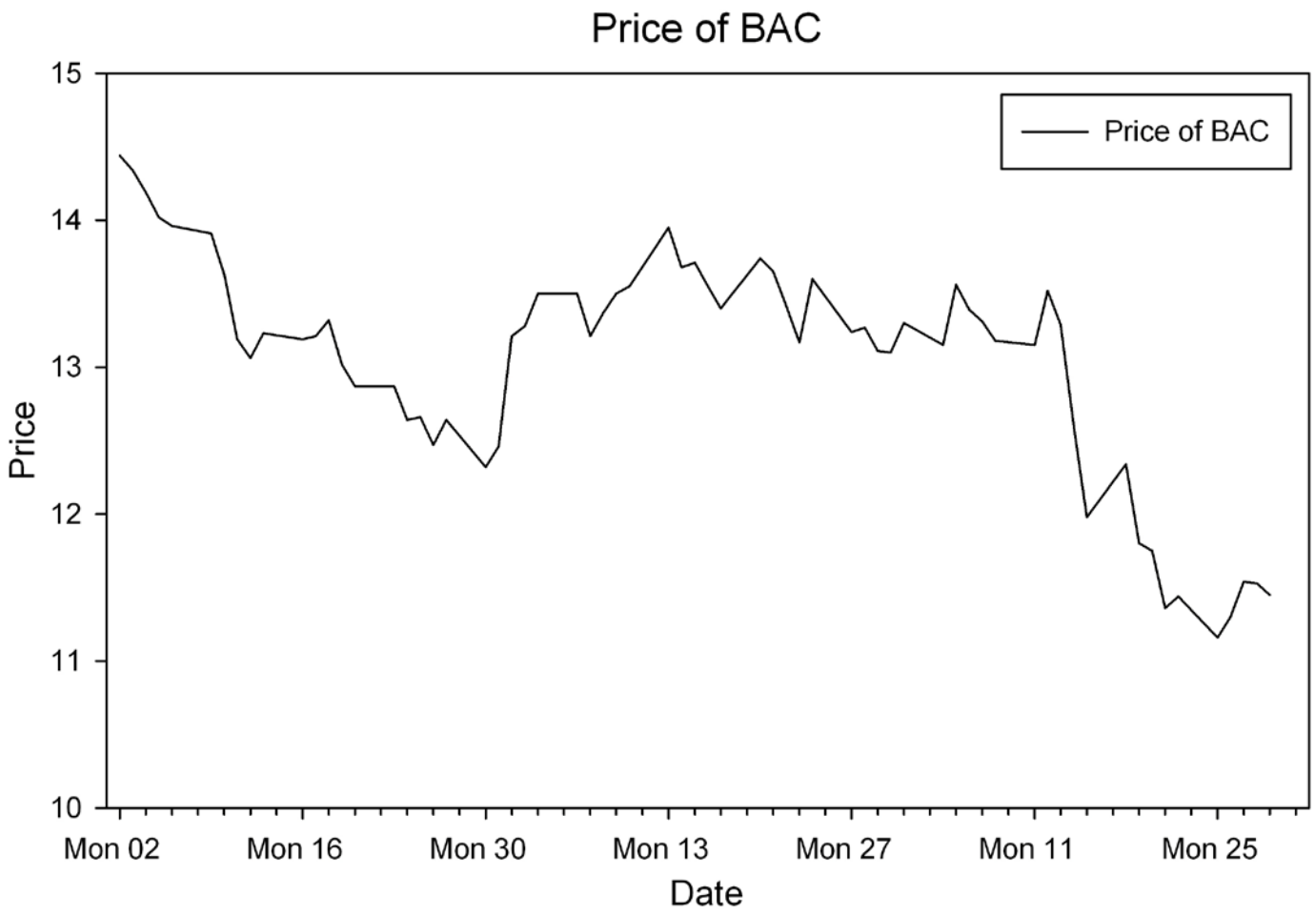
- Each share of stock represents a piece of a company
- The value of a stock is the current price at which people are willing to buy/sell shares
- These values increase/decrease based on supply and demand of the stock in question

Technical Analysis

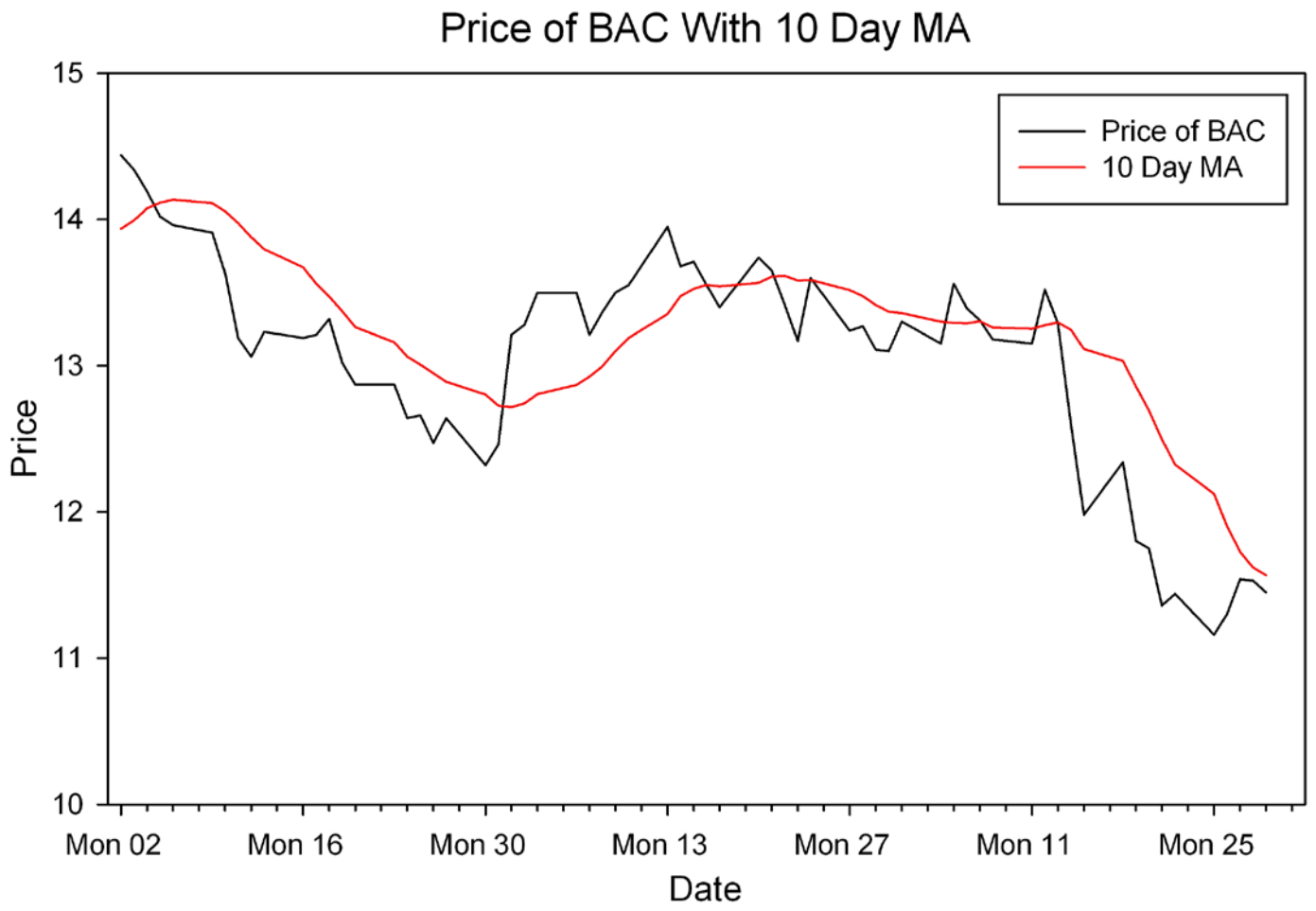
Carleton UNIVERSITY

- Aims to identify mispriced stocks
- Decisions based on historical price/volume information
- Many indicators to consider
  - Moving Averages
  - Volume Indices
  - Ease of Movement

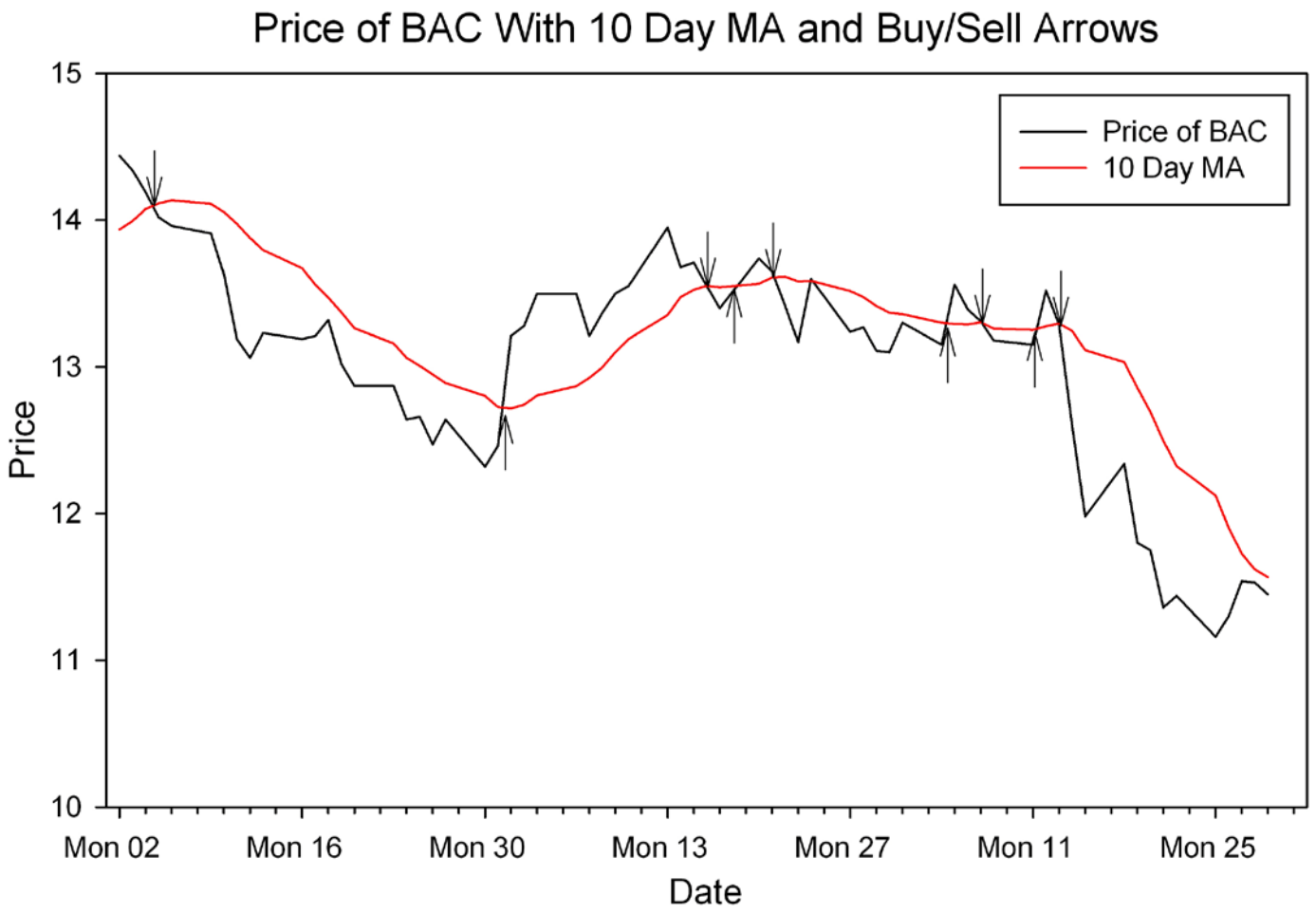
Technical Analysis



Technical Analysis



Technical Analysis

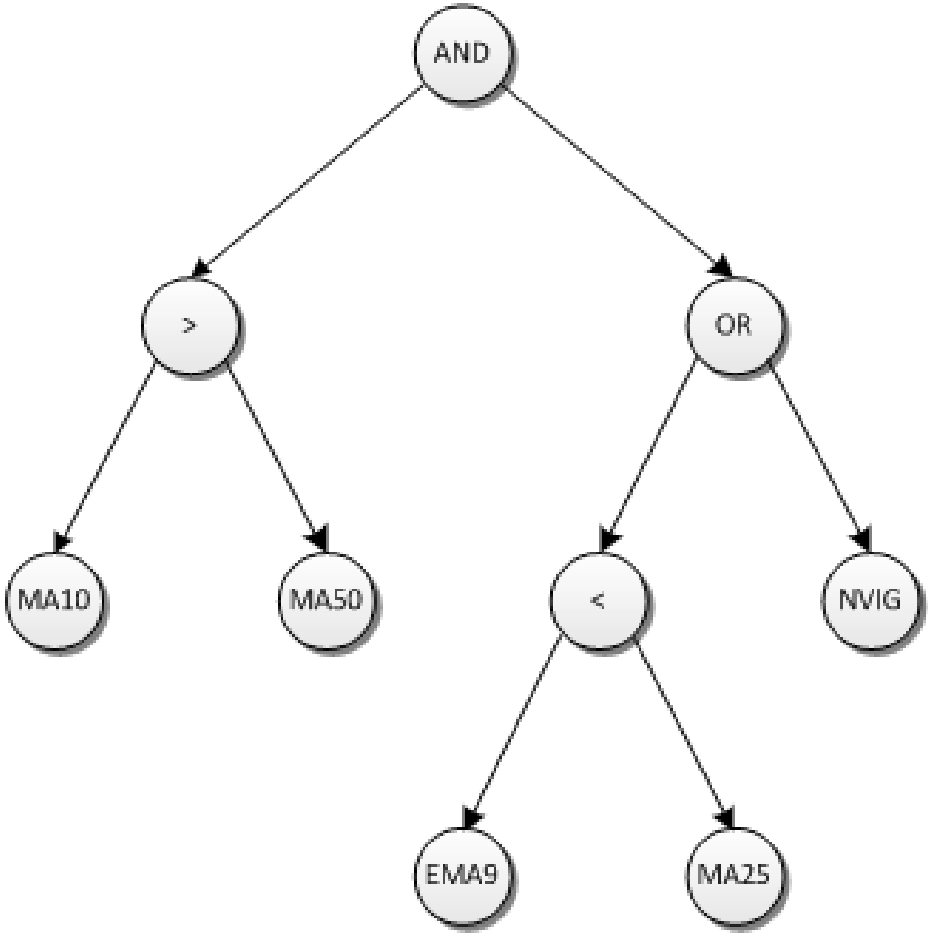


Genetic Programming

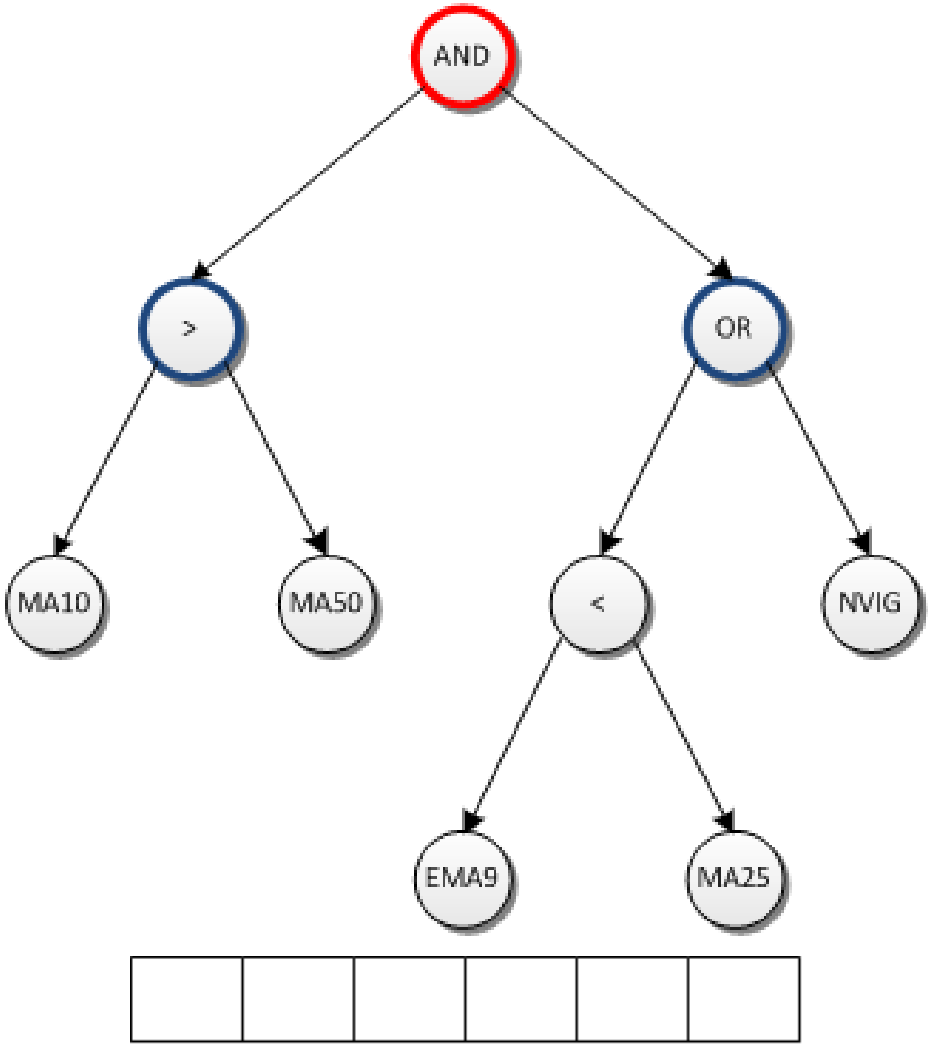
Carleton UNIVERSITY

- An approach that aims to evolve programs which solve a problem effectively
- Programs are represented as trees, with inputs as leafs and functions as inner nodes
- Each individual is tested against training data which determines the individual's fitness

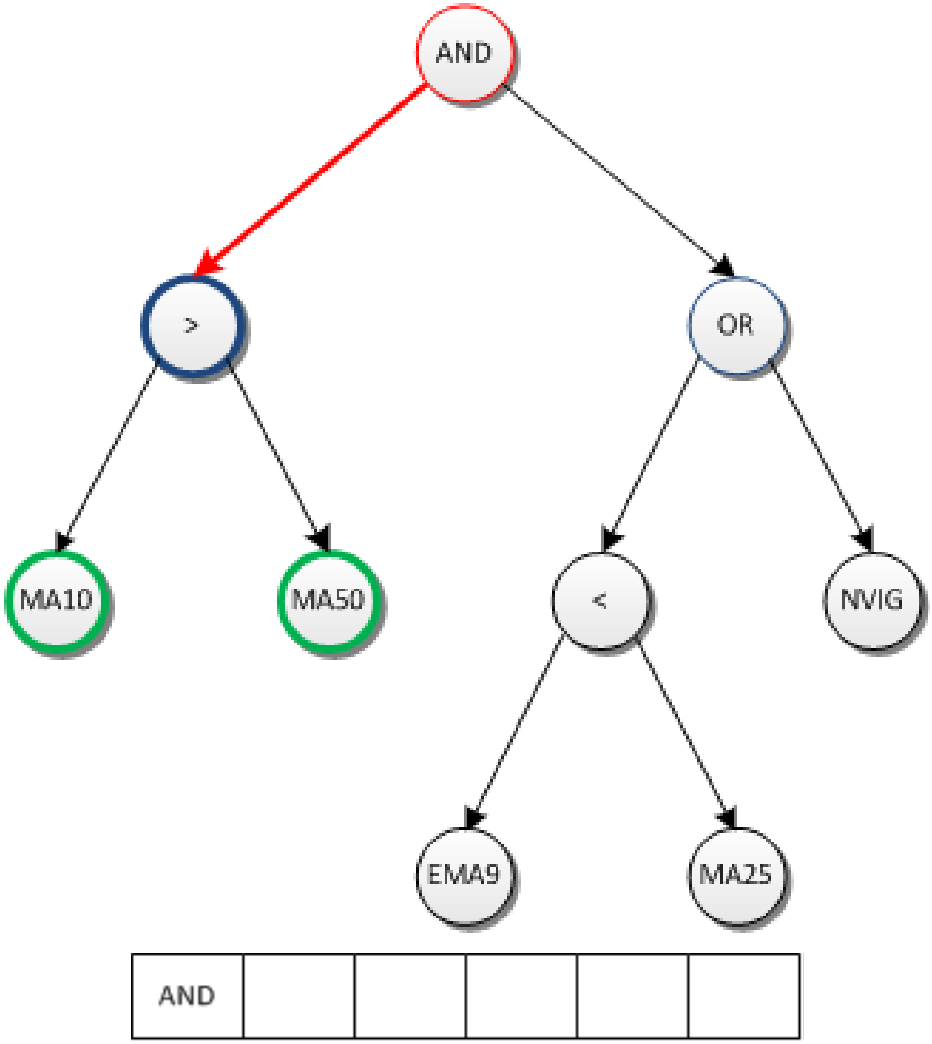
Example GP Tree



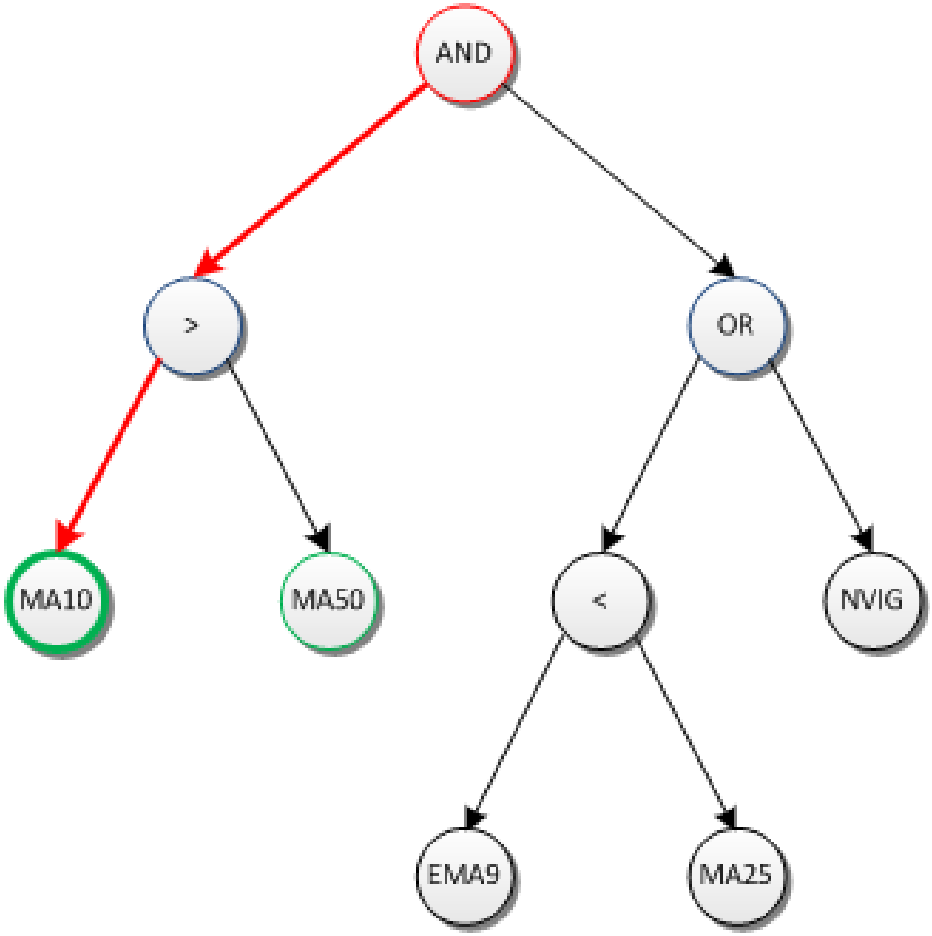
Example GP Tree



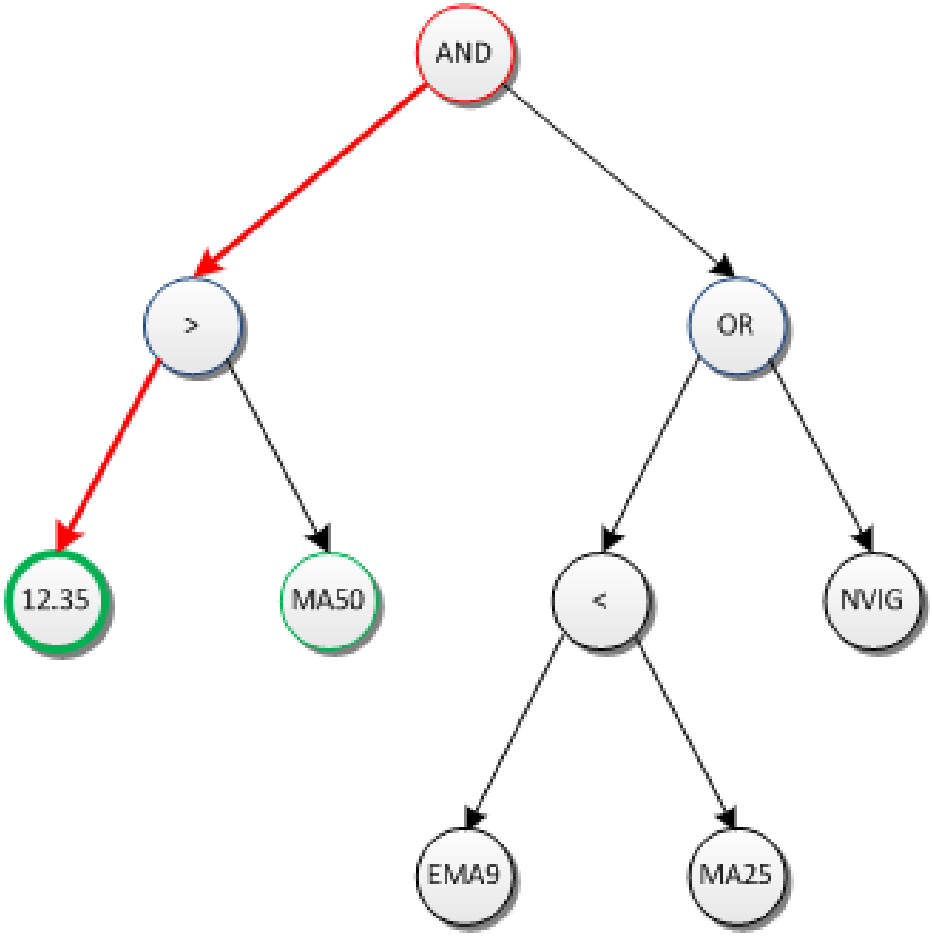
Example GP Tree



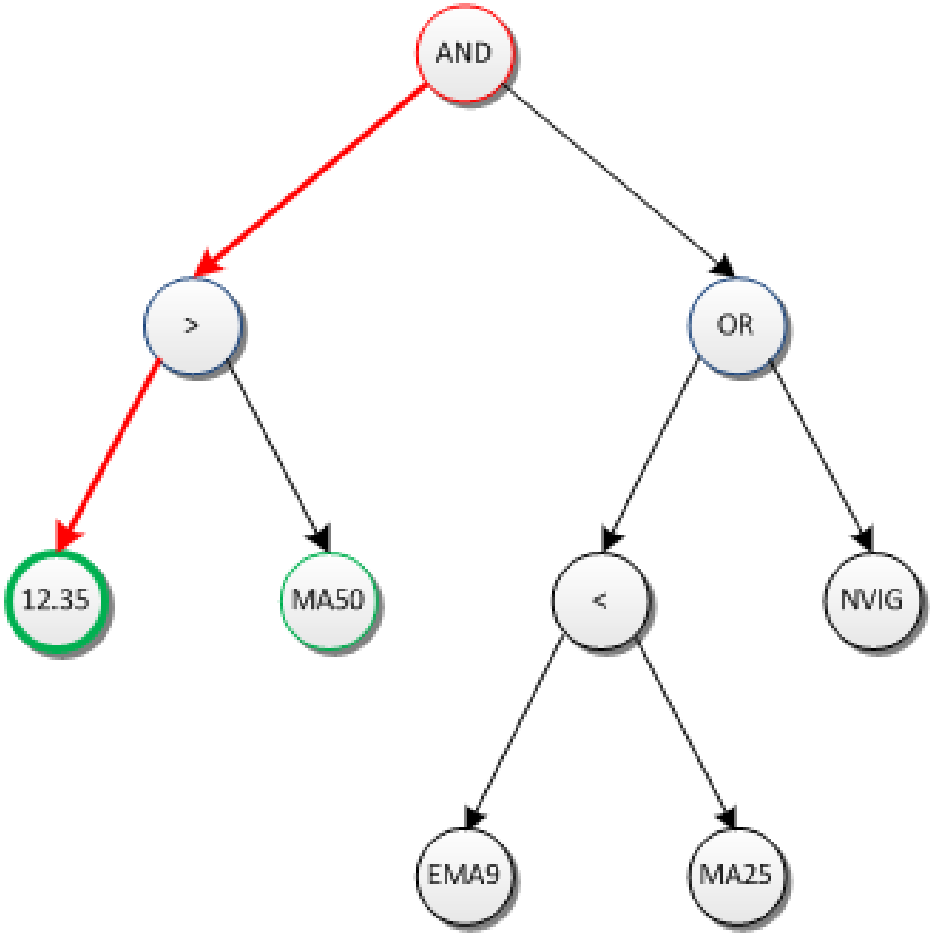
Example GP Tree



Example GP Tree

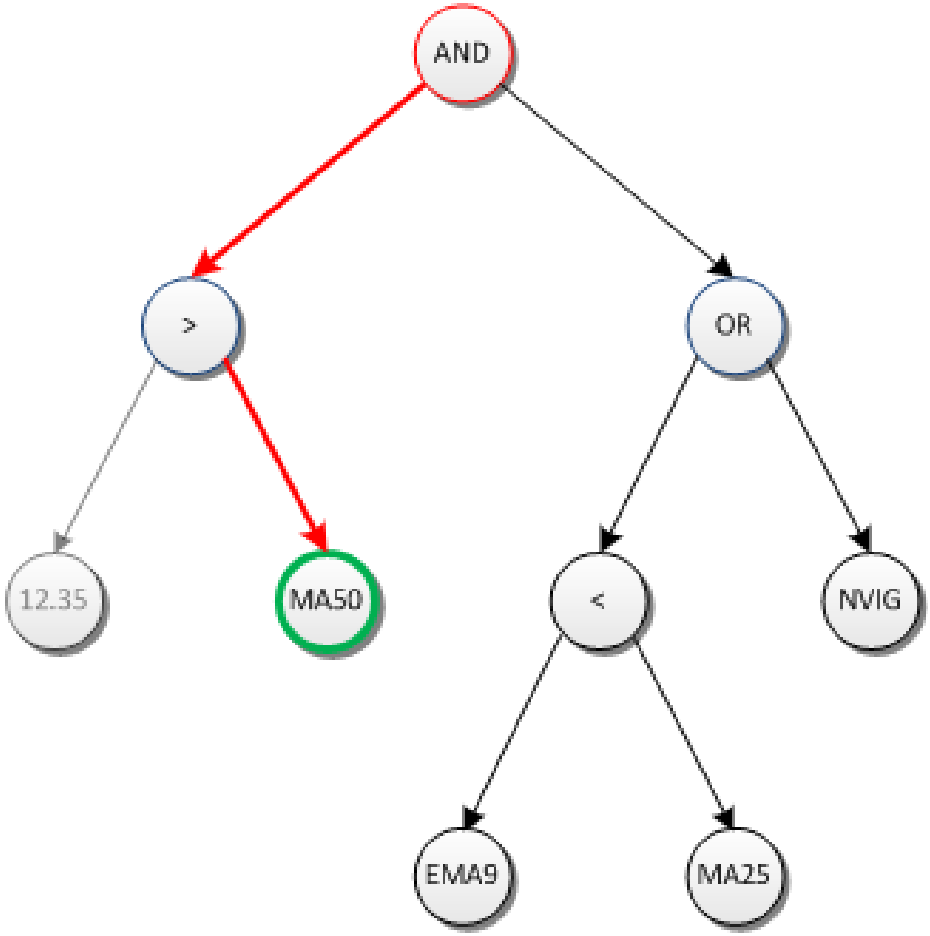


Example GP Tree



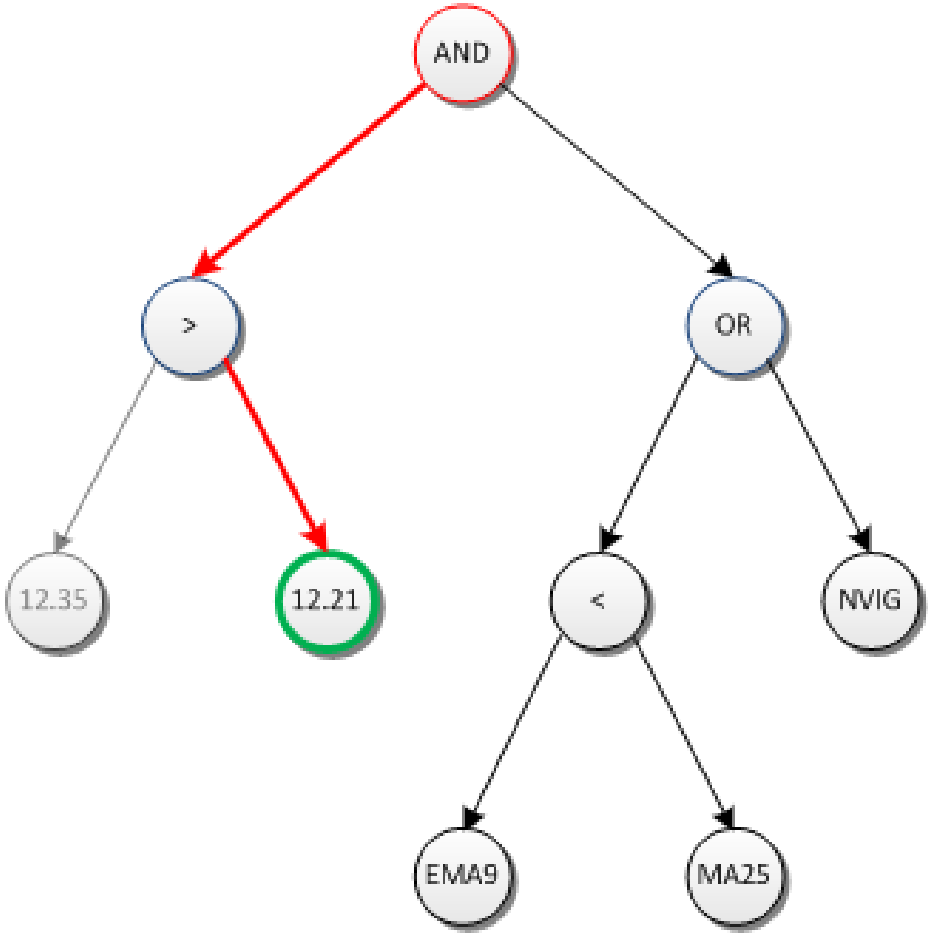
AND	>	12.35			

Example GP Tree



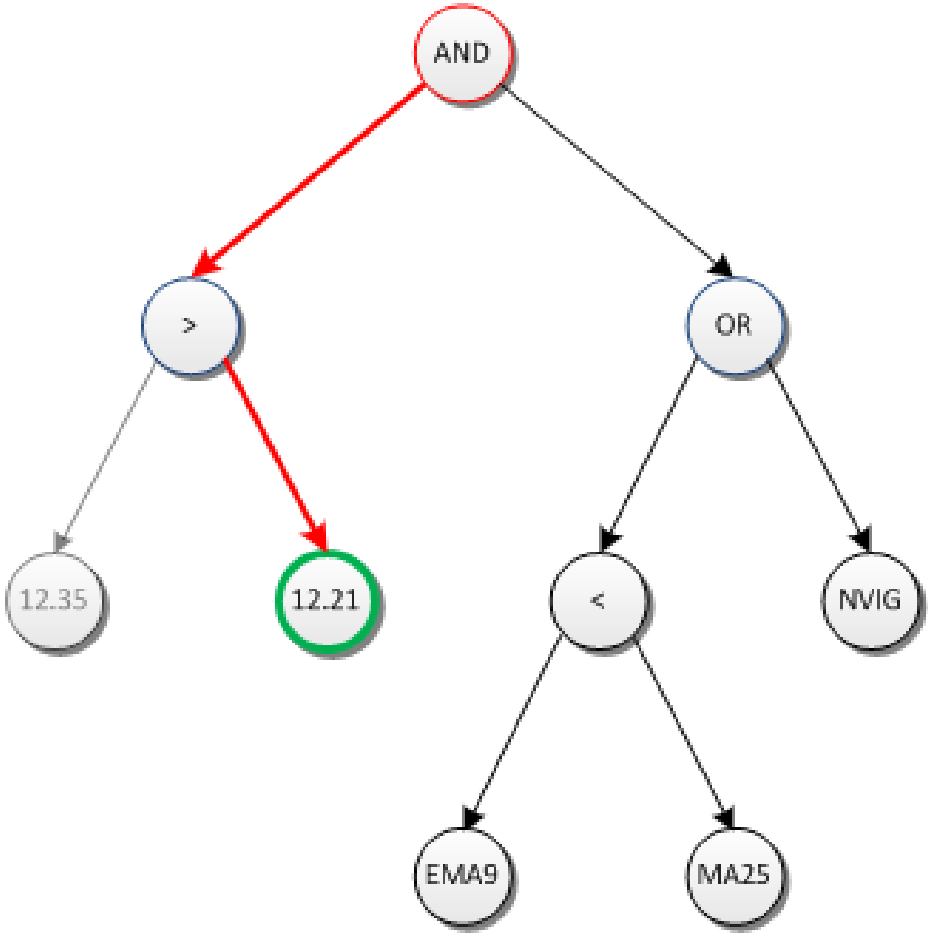
AND	>	12.35			

Example GP Tree



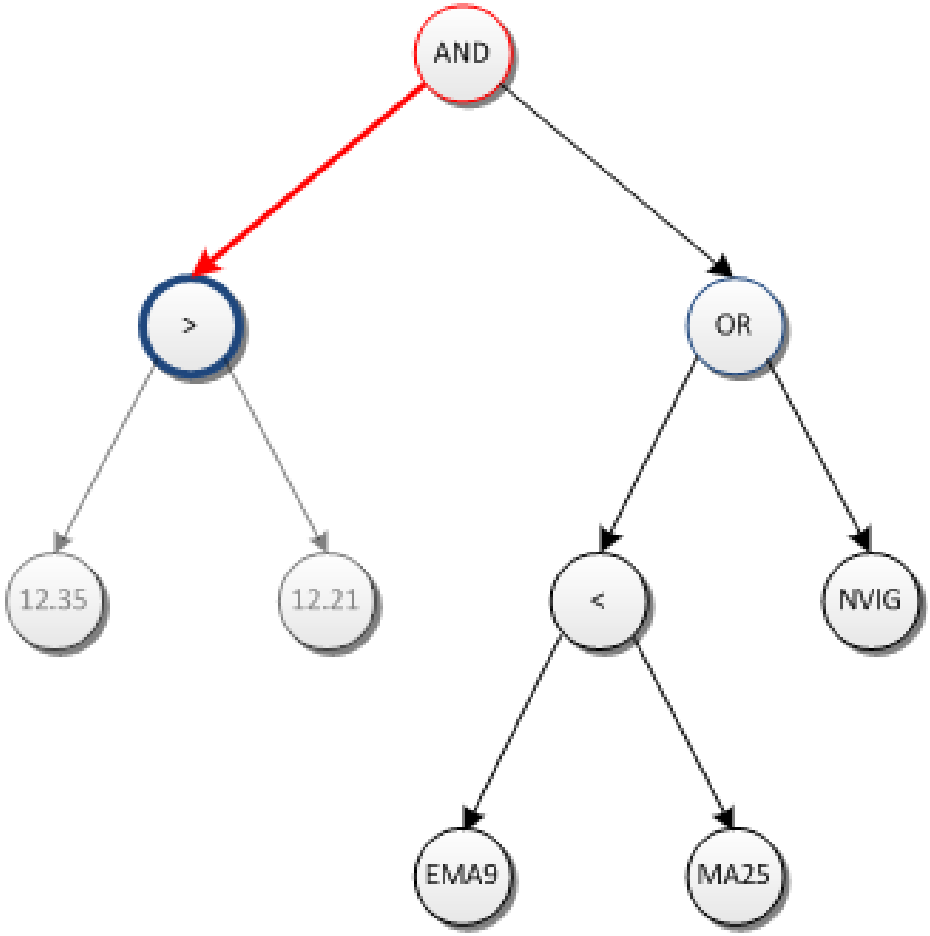
AND	>	12.35			

Example GP Tree



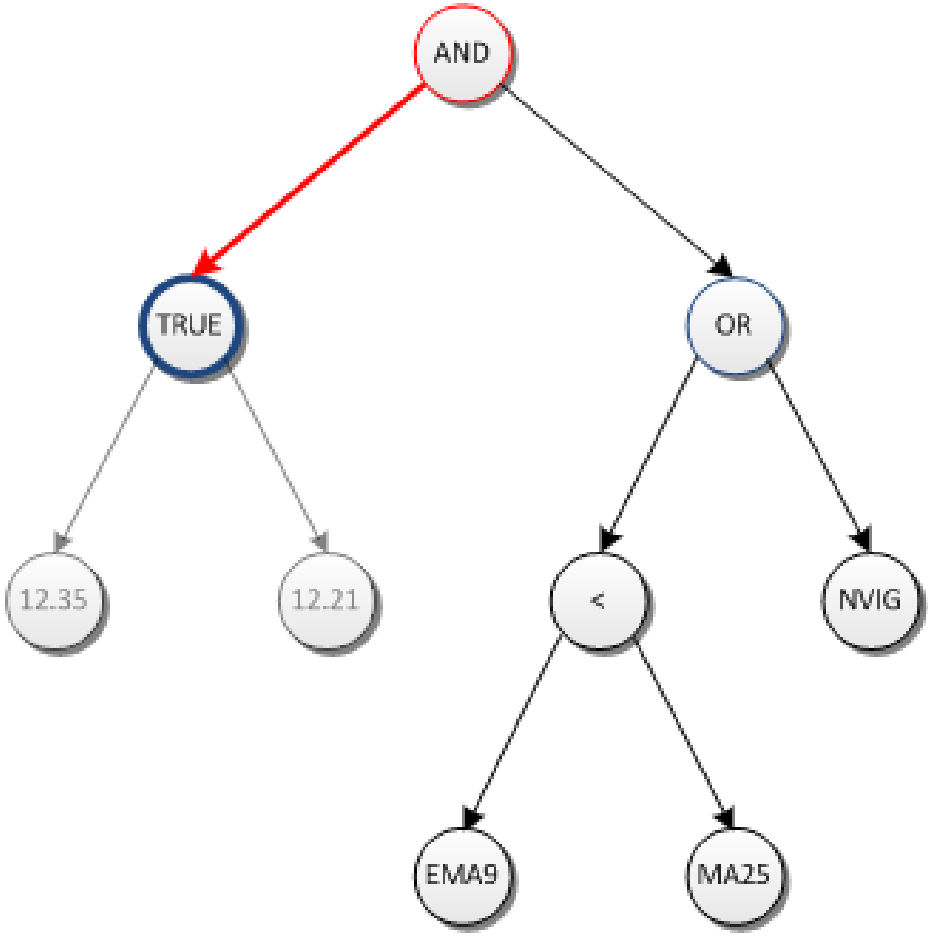
AND	>	12.35	12.21		

Example GP Tree

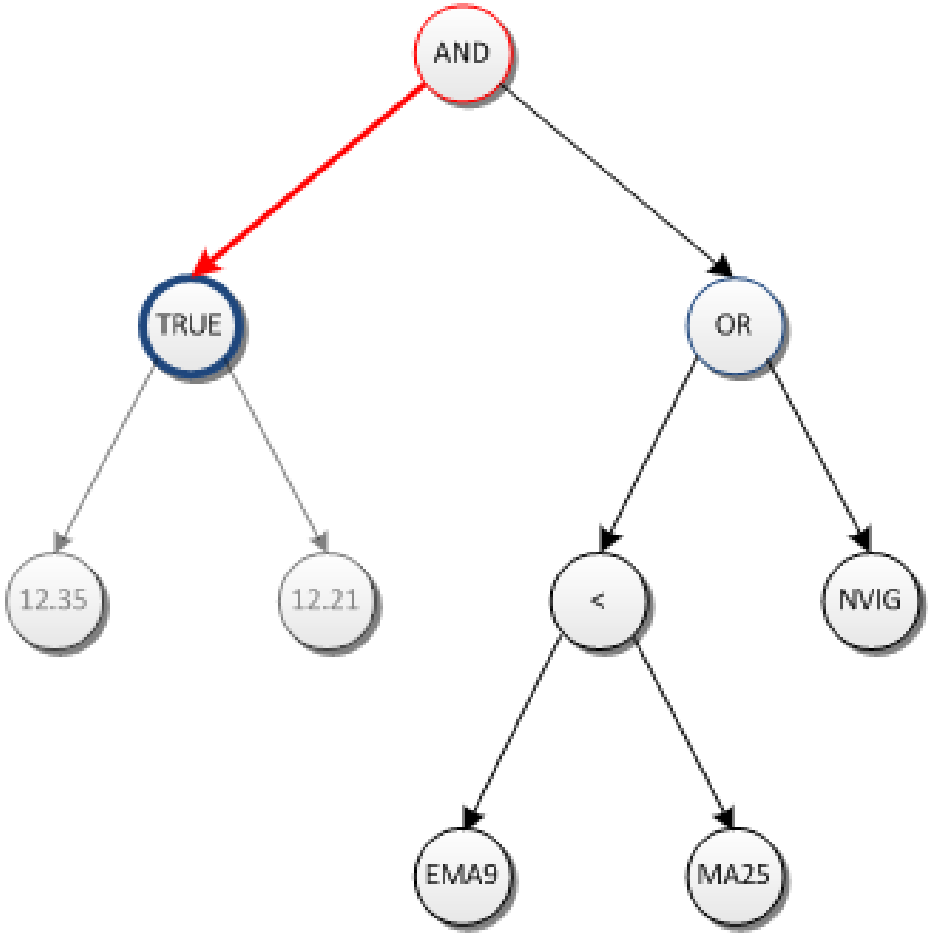


AND	>	12.35	12.21		

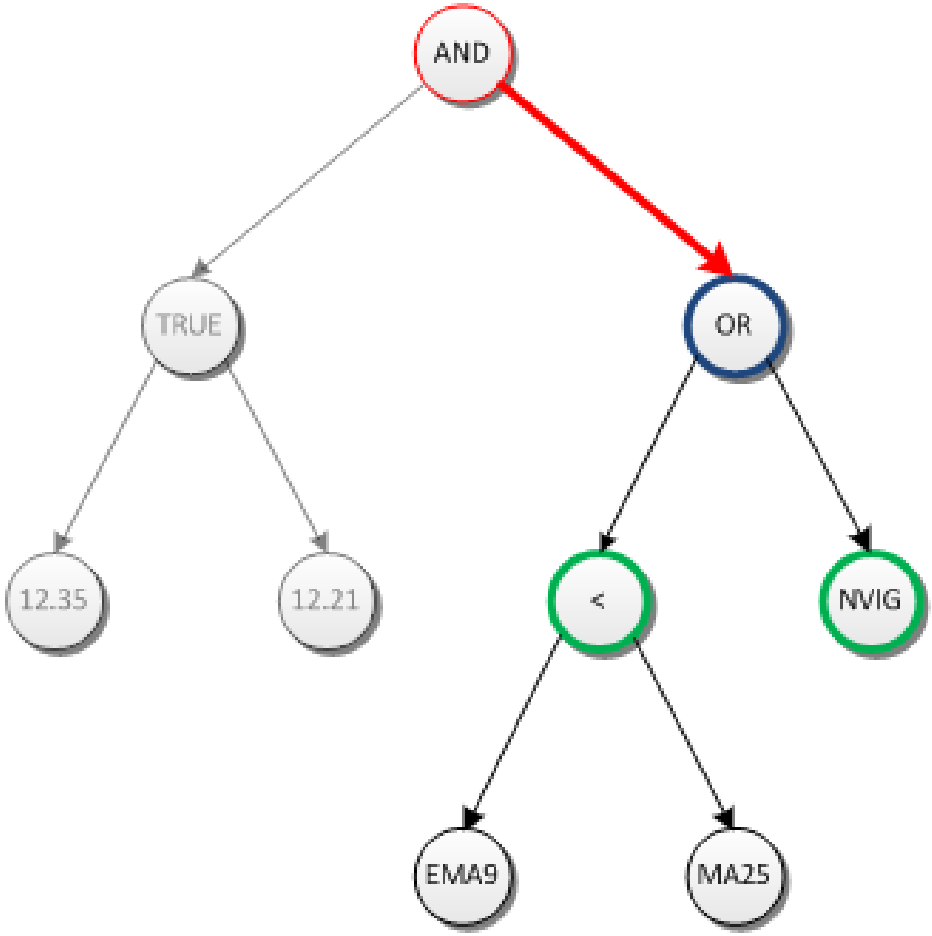
Example GP Tree



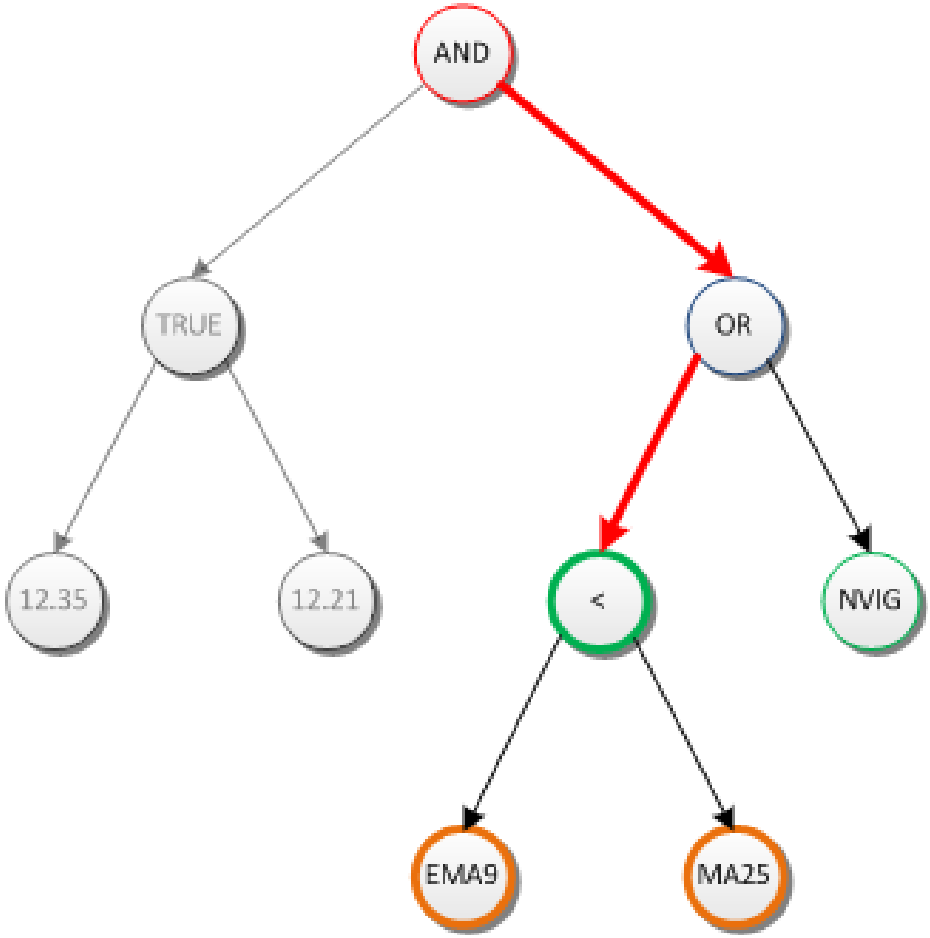
Example GP Tree



Example GP Tree

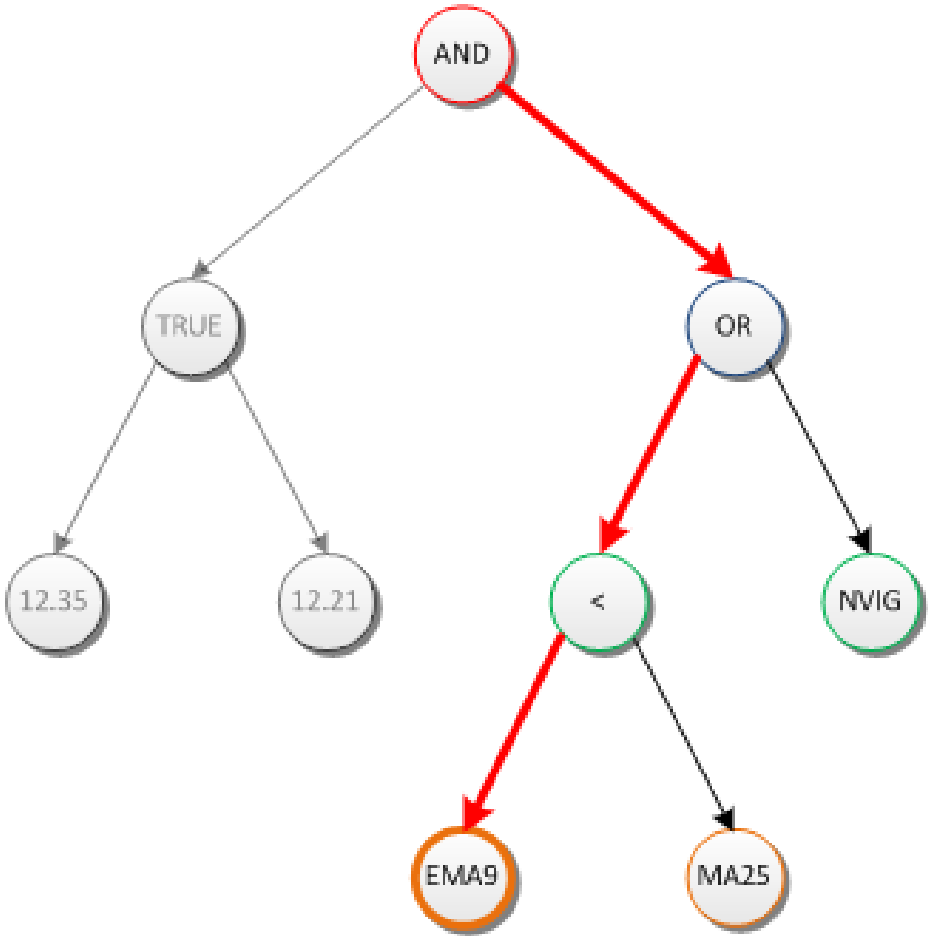


Example GP Tree



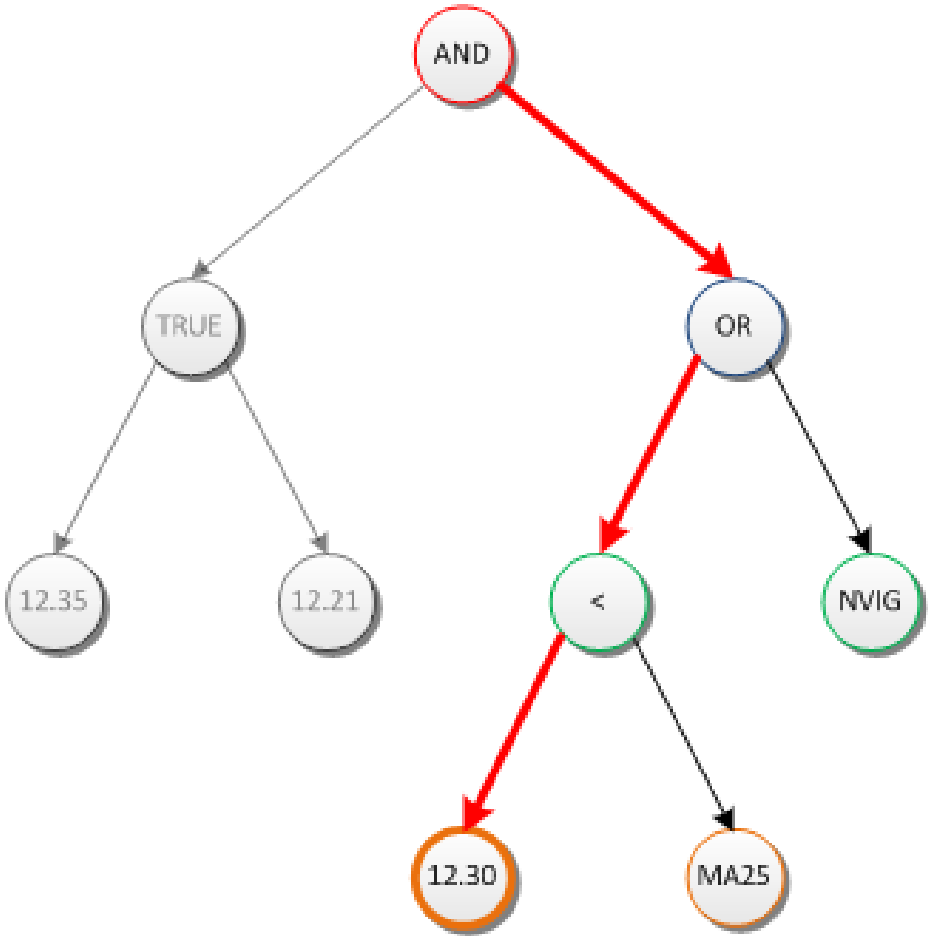
AND	TRUE	OR			

Example GP Tree



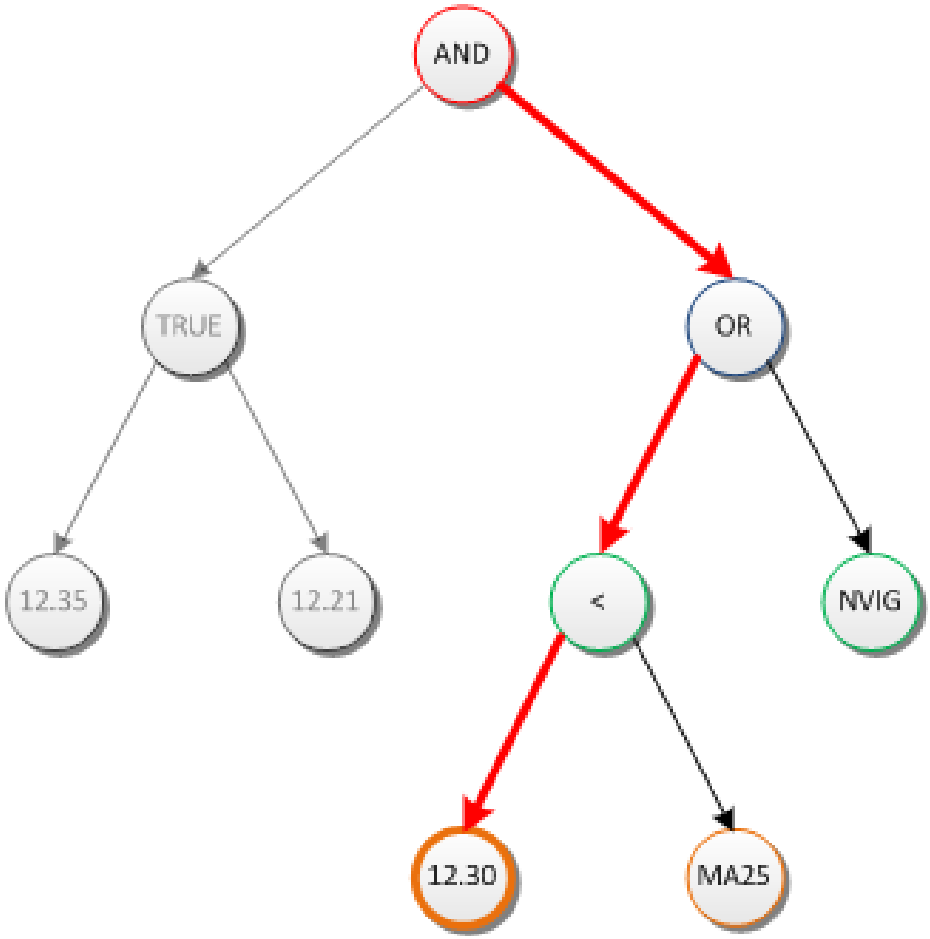
AND	TRUE	OR	<		

Example GP Tree



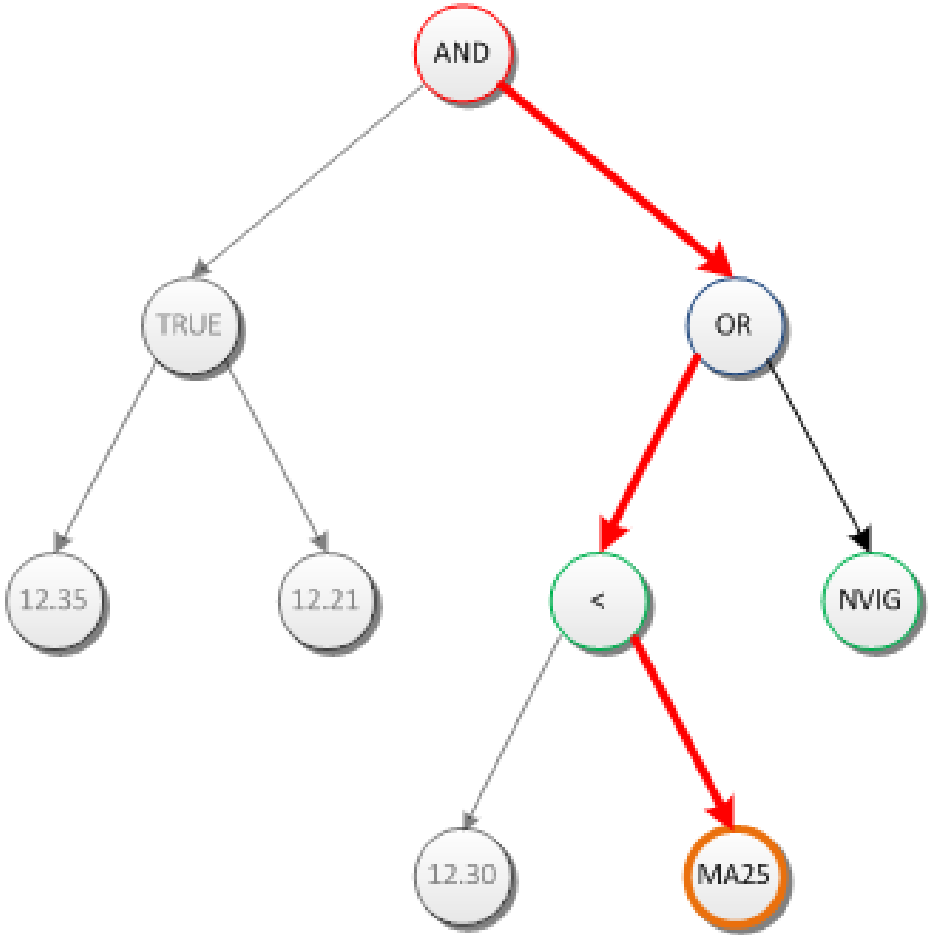
AND	TRUE	OR	<		

Example GP Tree



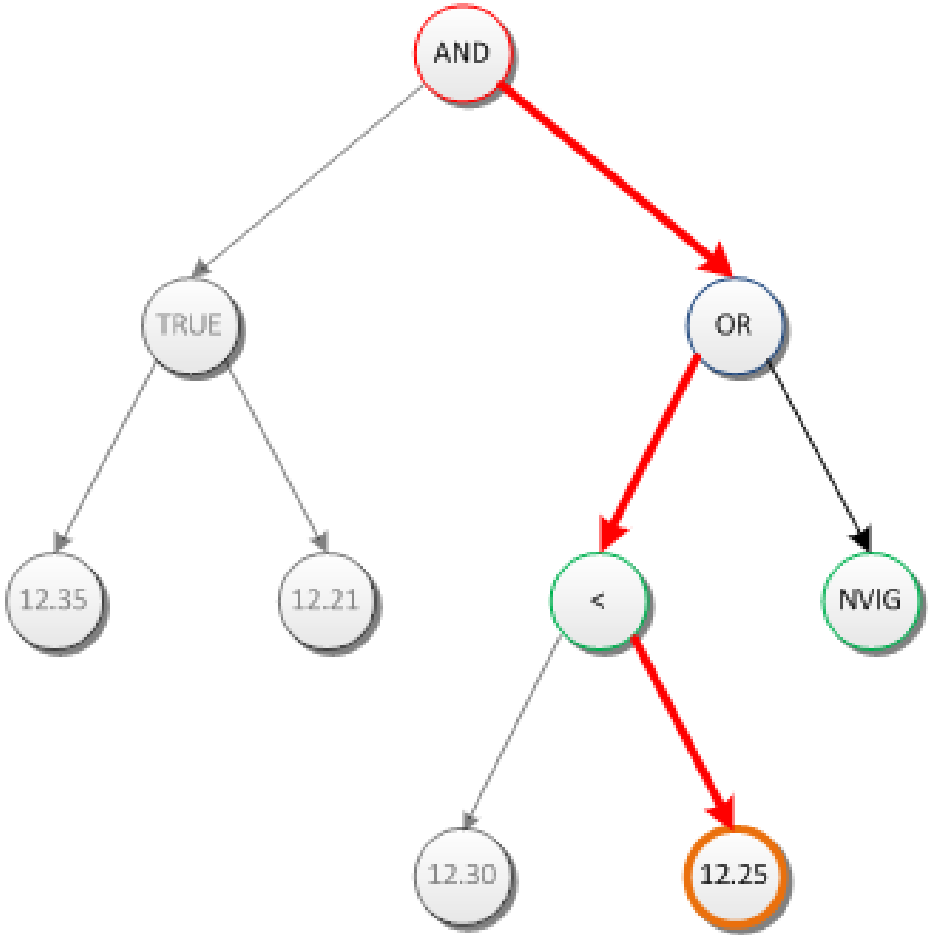
AND	TRUE	OR	<	12.30	

Example GP Tree



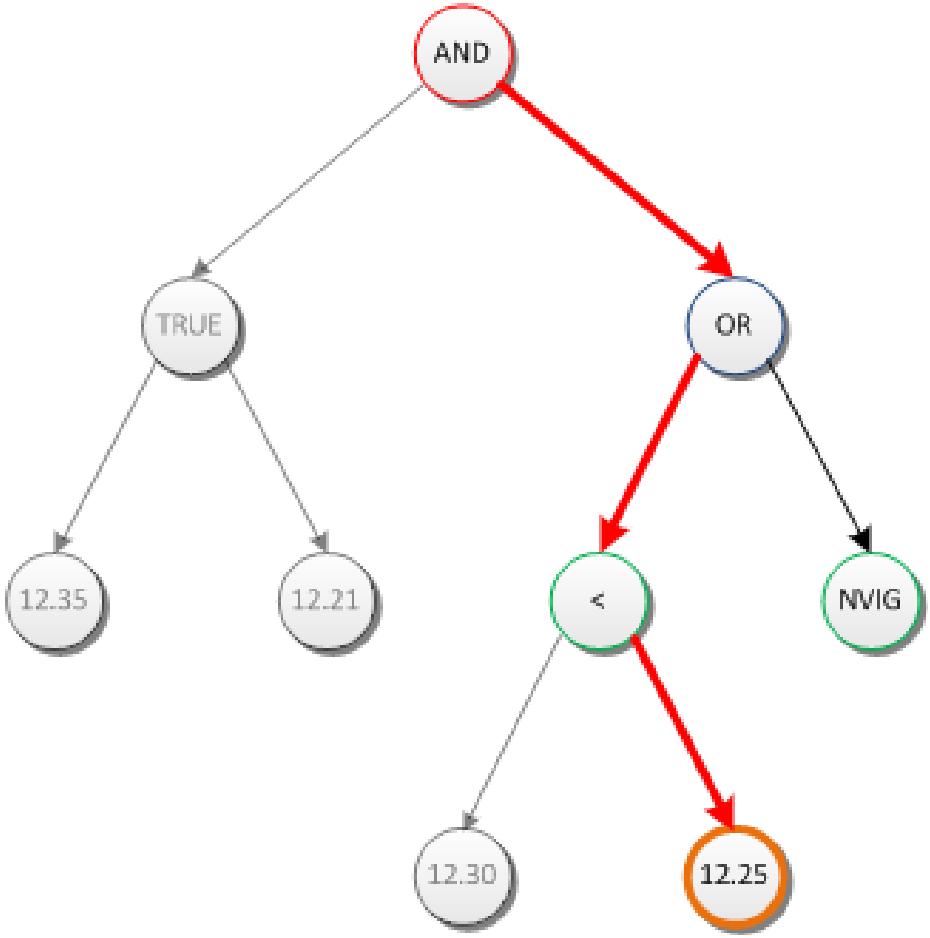
AND	TRUE	OR	<	12.30	

Example GP Tree



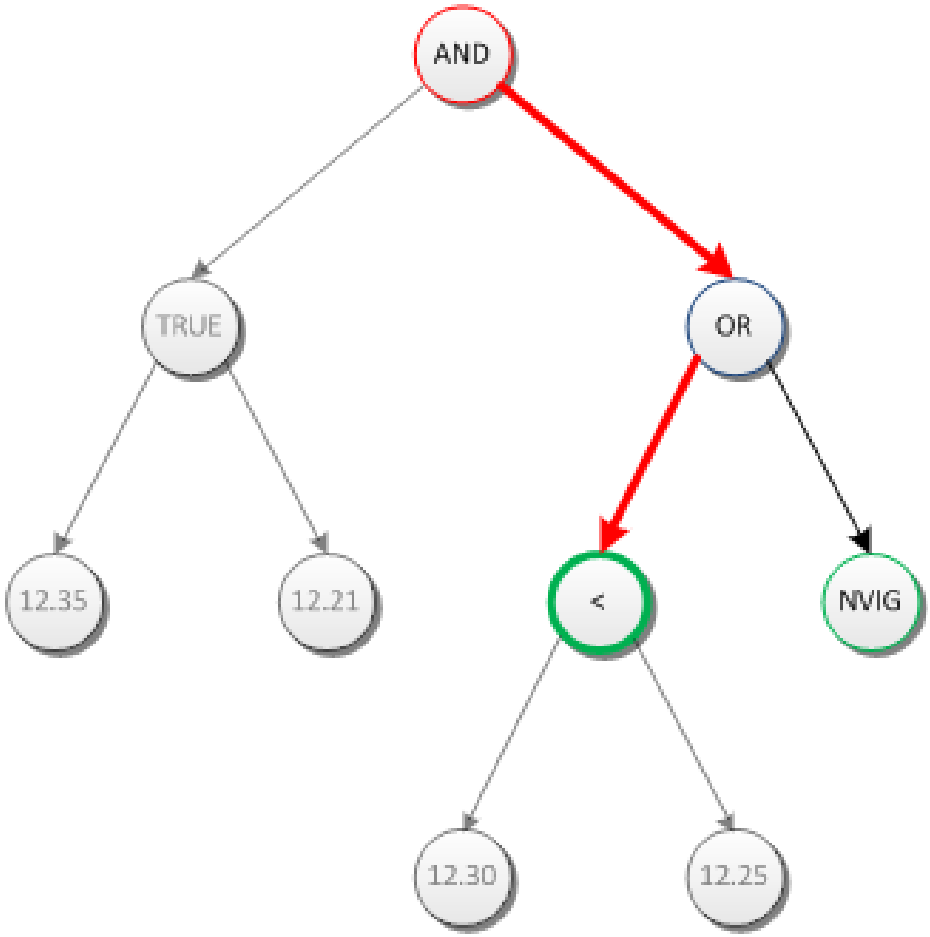
AND	TRUE	OR	<	12.30	

Example GP Tree



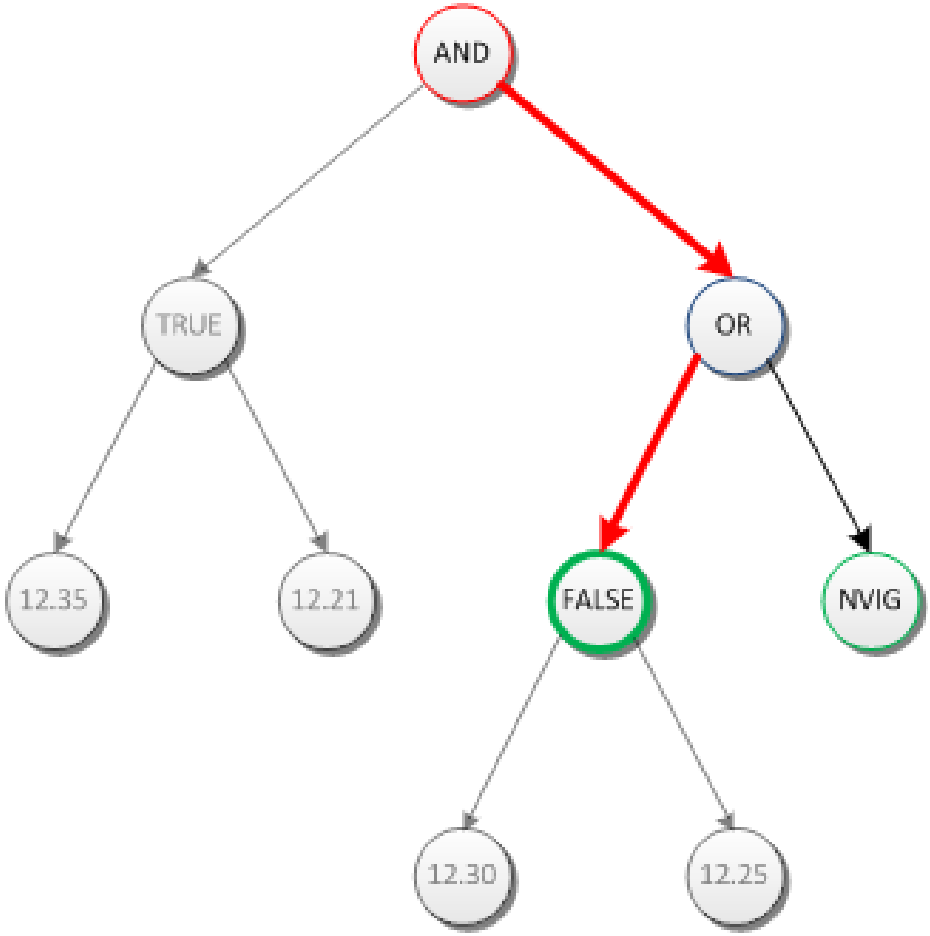
AND	TRUE	OR	<	12.30	12.25

Example GP Tree



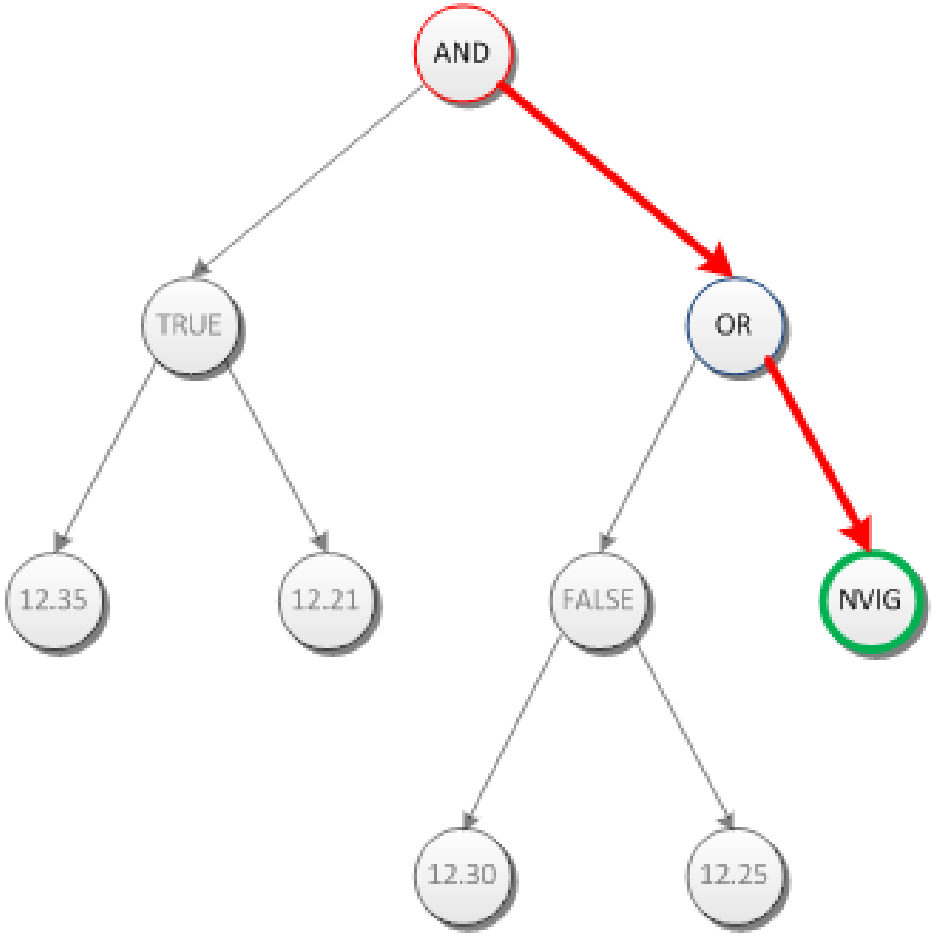
AND	TRUE	OR	<	12.30	12.25

Example GP Tree

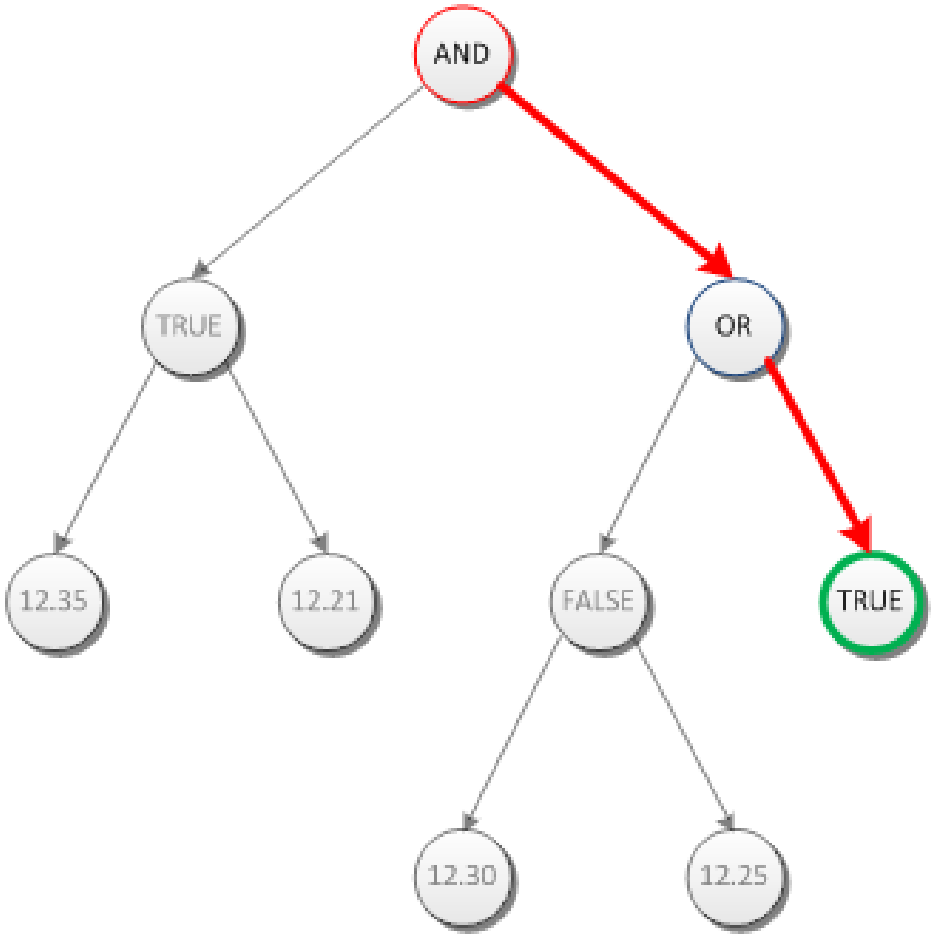


AND	TRUE	OR	FALSE		

Example GP Tree

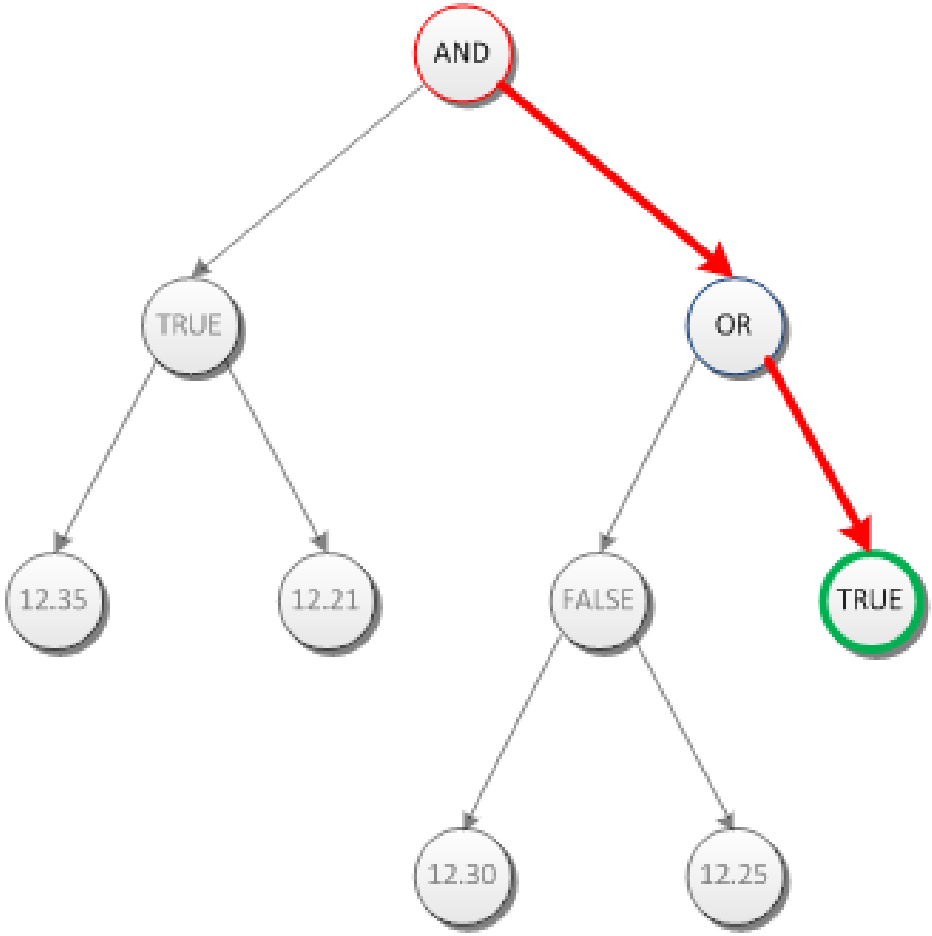


Example GP Tree



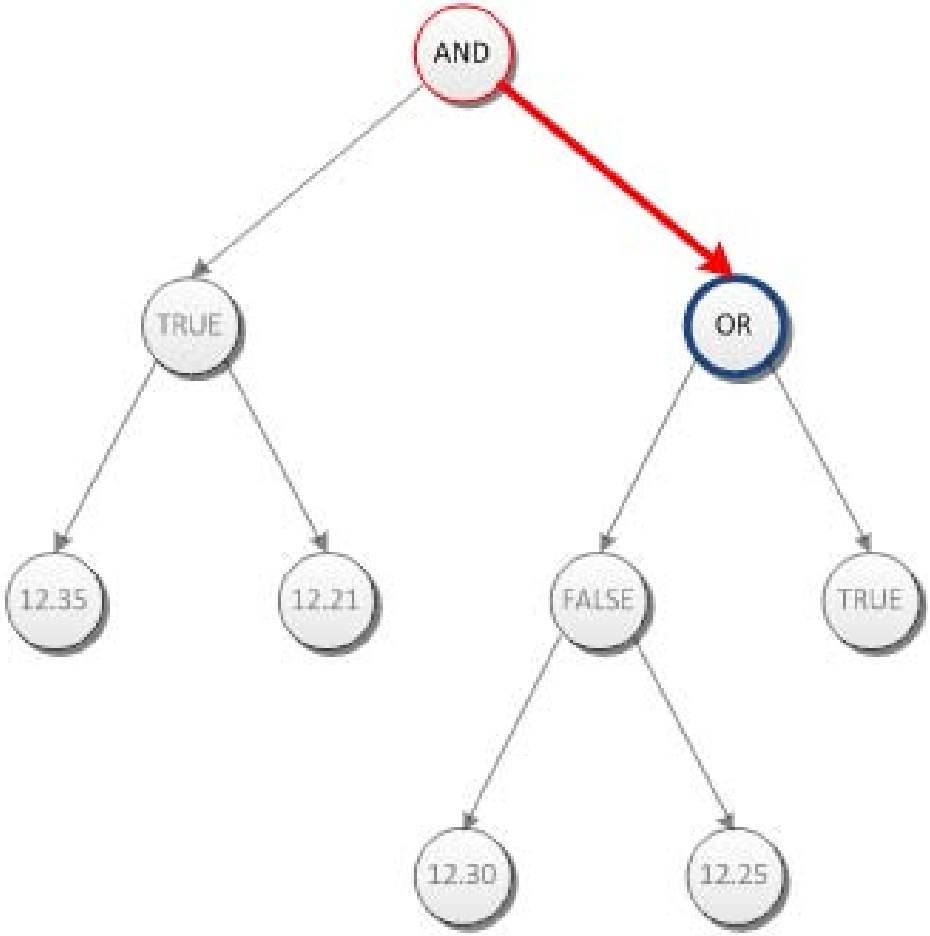
AND	TRUE	OR	FALSE		

Example GP Tree



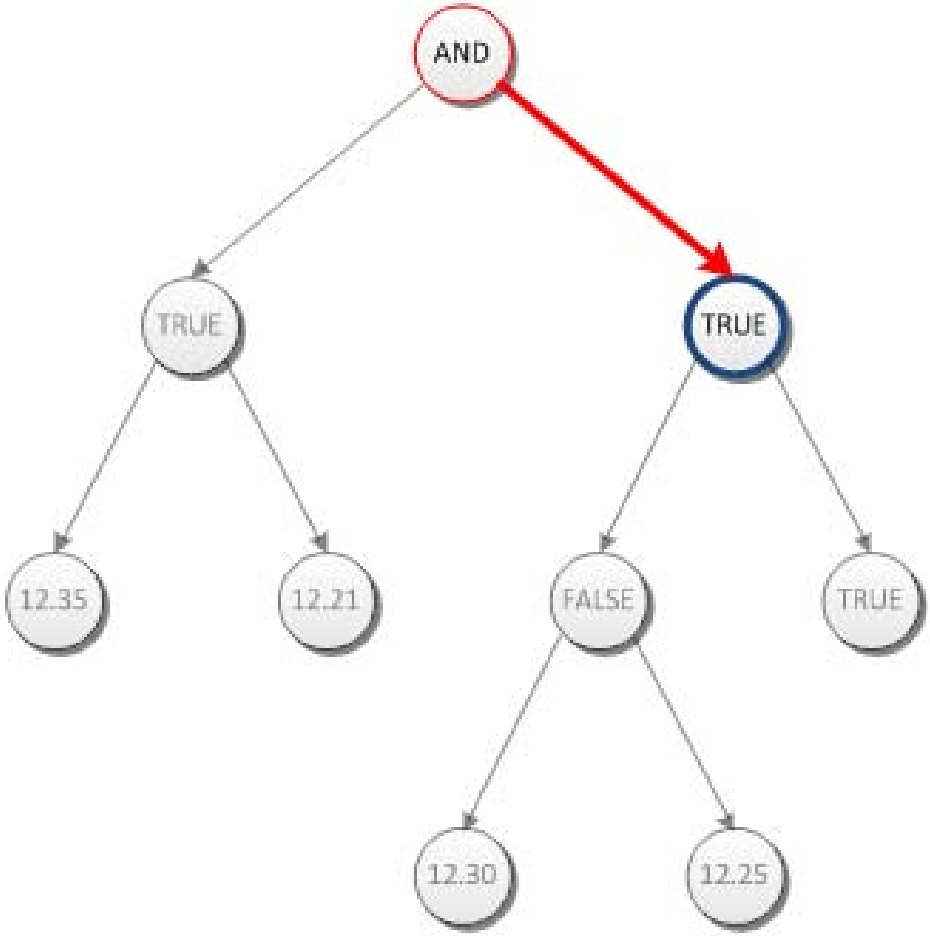
AND	TRUE	OR	FALSE	TRUE	

Example GP Tree

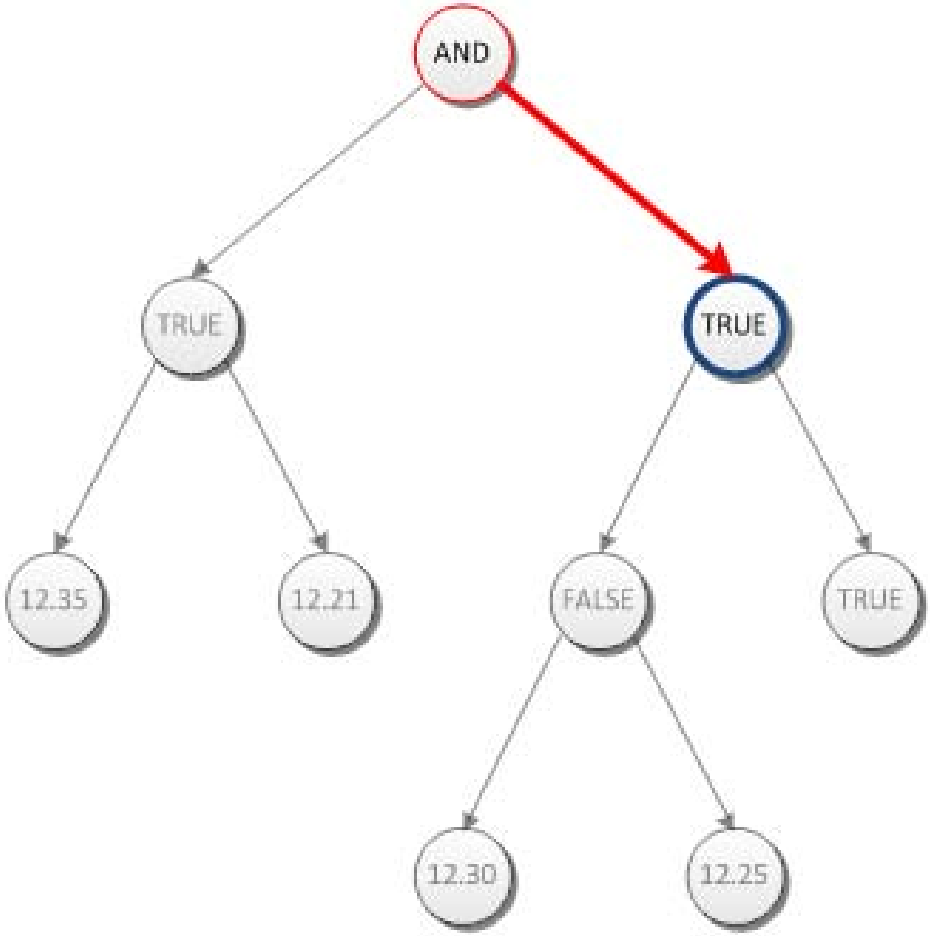


AND	TRUE	OR	FALSE	TRUE	

Example GP Tree

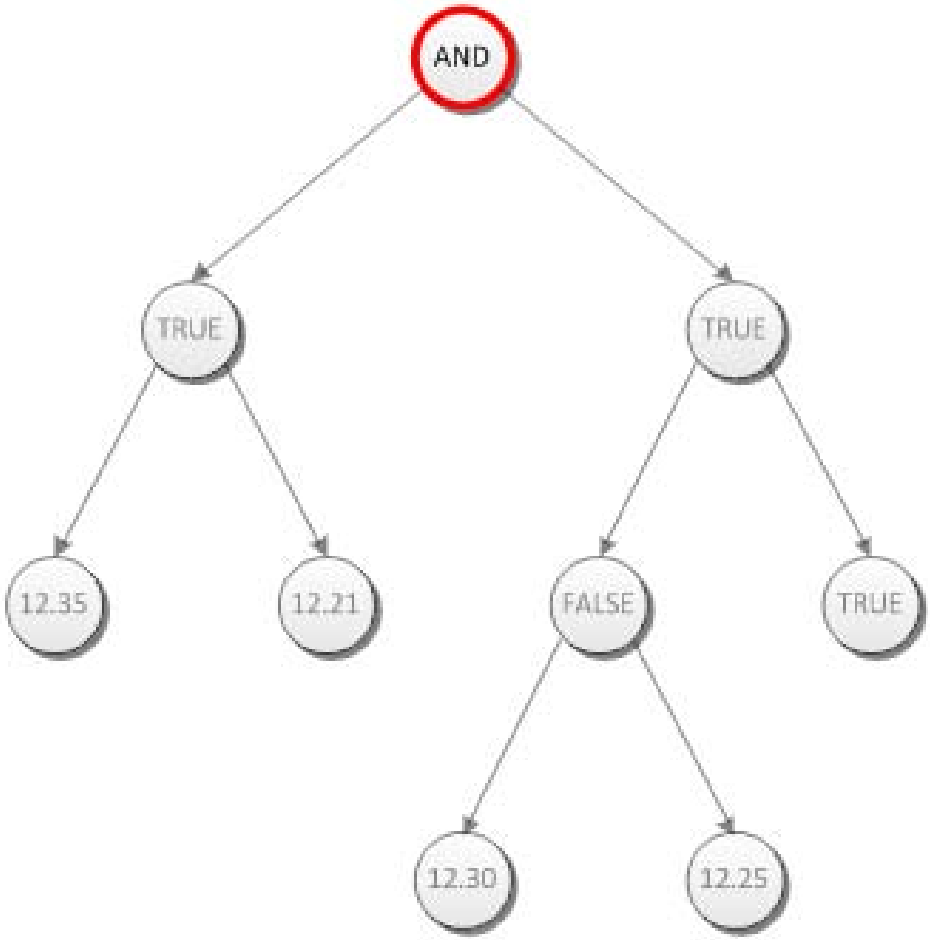


Example GP Tree



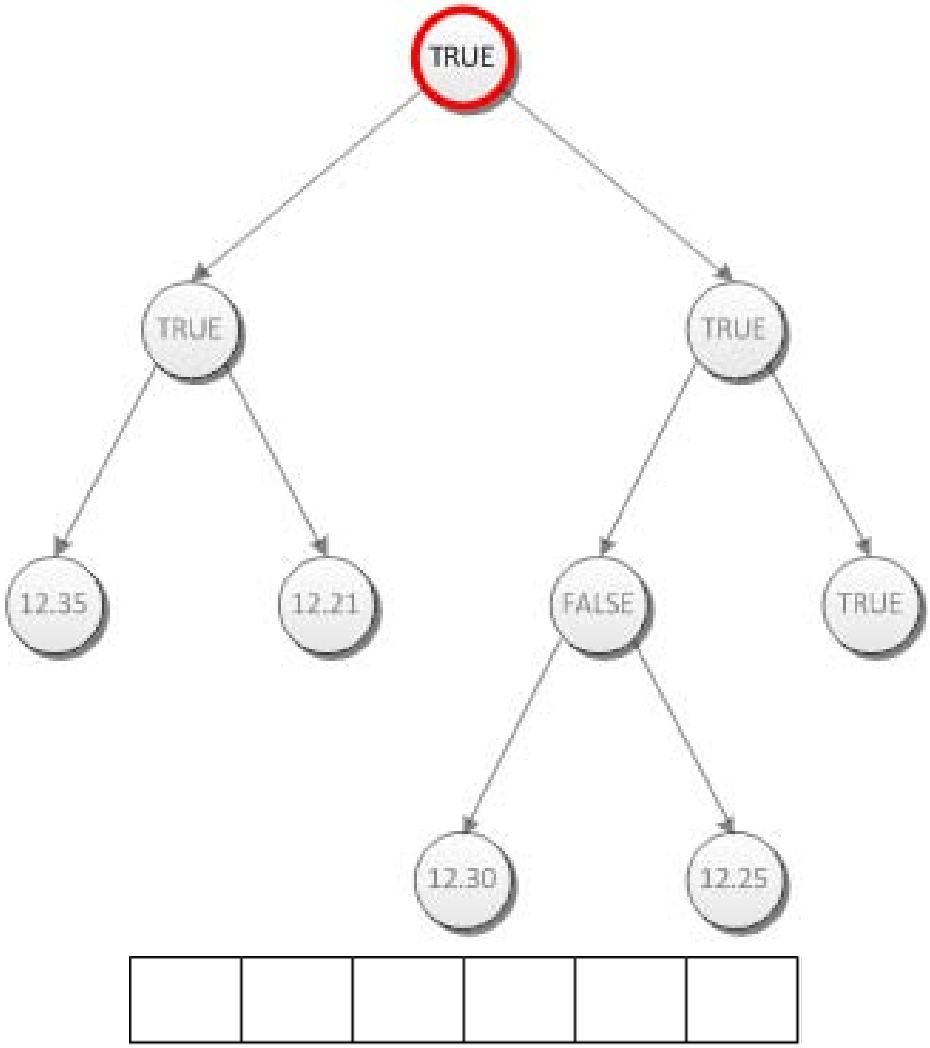
AND	TRUE	TRUE			

Example GP Tree

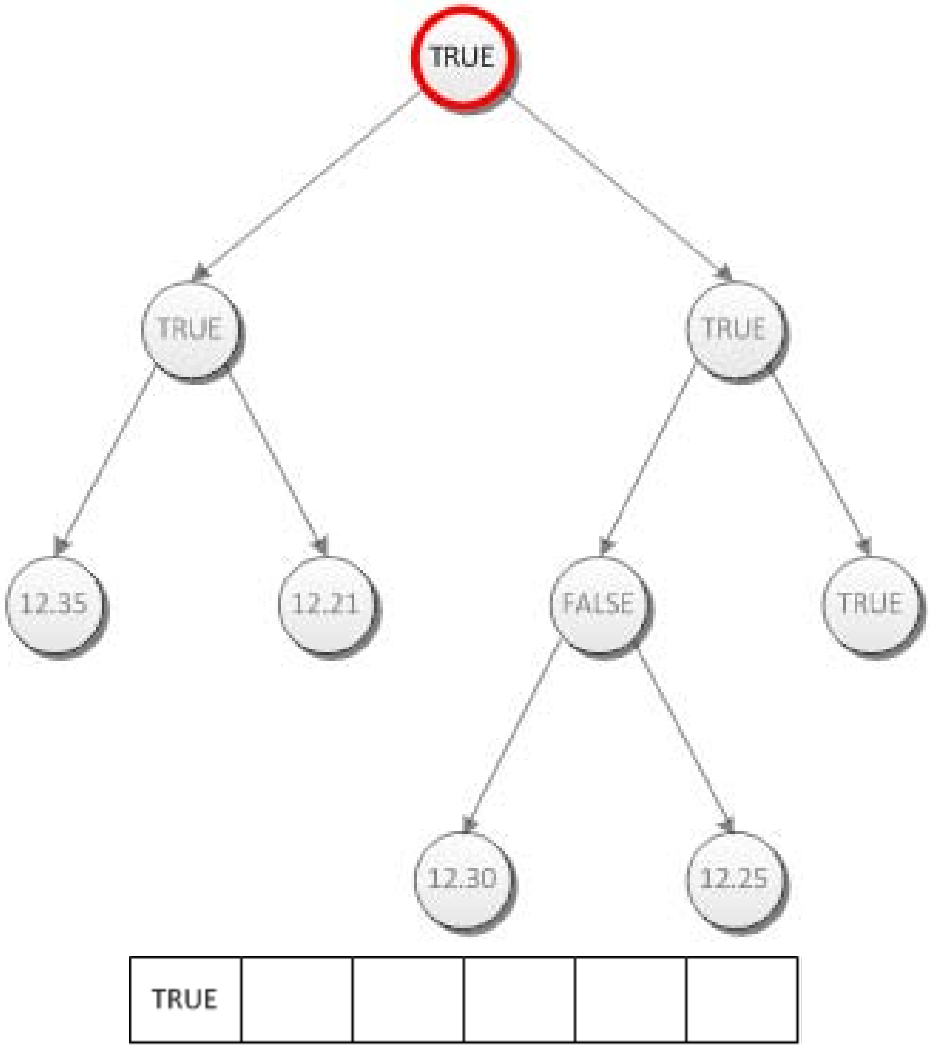


AND	TRUE	TRUE			

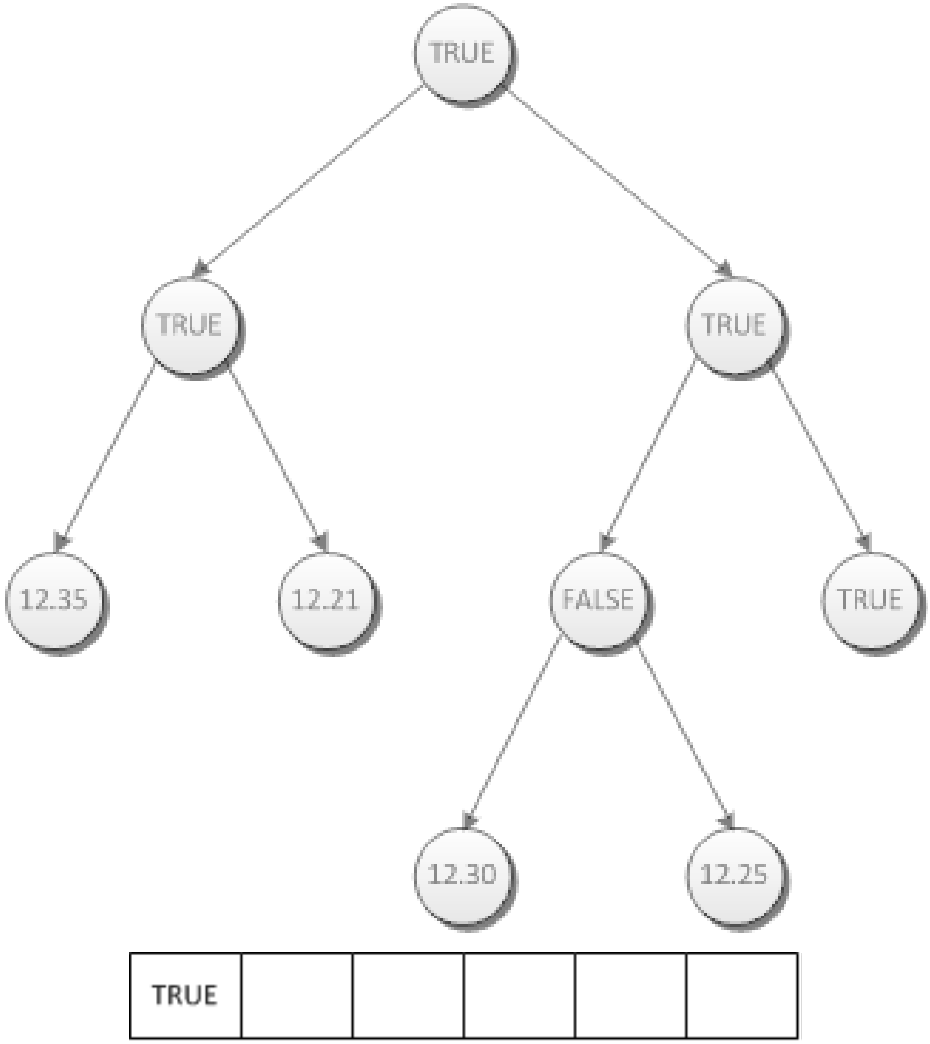
Example GP Tree



Example GP Tree



Example GP Tree



GP Details

- Five Boolean Functions:
  - AND, OR, NOT, >, <
- 17 Money Value Indicators:
  - MAs, EMAs, Closing/Typical Prices
- 14 Boolean Value Indicators:
  - Ease of movement, MACD, etc.

GP Details

Carleton UNIVERSITY

- Each individual consists of two trees
  - A buy signal and sell signal tree
- Tree values for a given day determine action
- A given amount of money is allocated for each stock
- The stocks are traded for a number of days
- Fitness is calculated based on ROI

GP Decision Matrix

Carleton UNIVERSITY

		Buy Tree	
		True	False
Sell Tree	True	No Action	Sell All Shares
	False	Buy Max. Shares	No Action

Why Use GPUs

Carleton UNIVERSITY

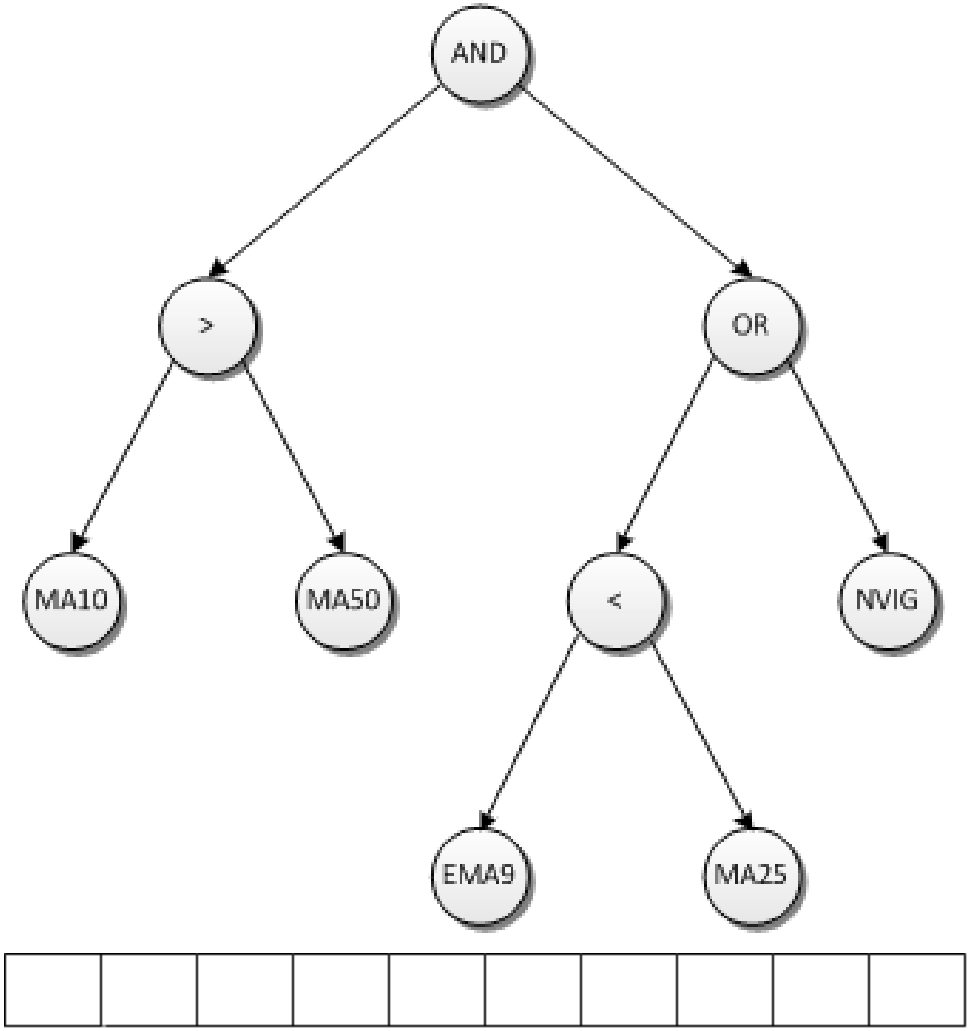
- Some typical parameters
  - 100 000 Individuals, 100 Stocks, 1000 Days
- Each individual's trees must be evaluated for each stock/day combination
  - 20 000 000 000 tree evaluations per generation
- With GPU these evaluations can be massively parallelized, resulting in extremely large speedups

GP On GPU Devices

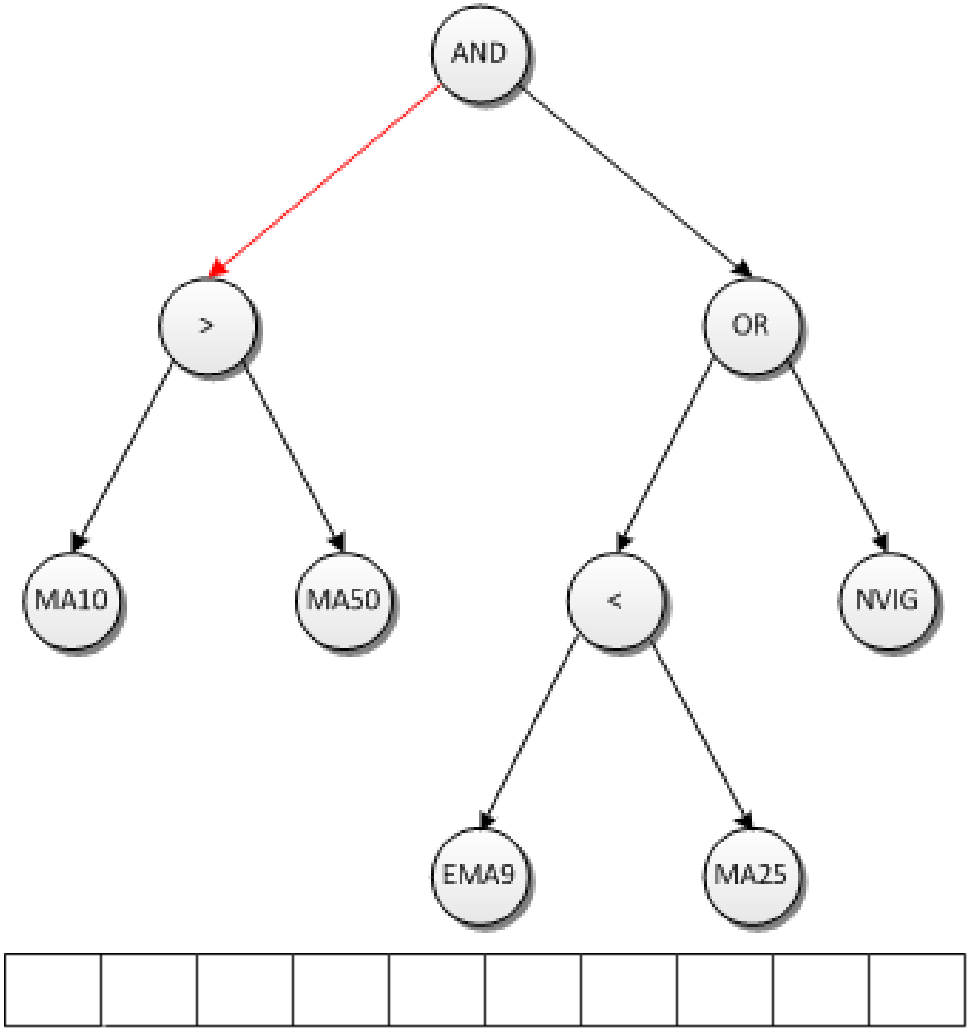
Carleton UNIVERSITY

- Until recently, there was no recursion support within CUDA (no execution stack)
- Tree-based GP cannot run without a stack
- The solution:
  - Implement a stack on the GPU
  - Convert tree-based individuals into RPN
  - Parse these 'trees' using the stack interpreter program

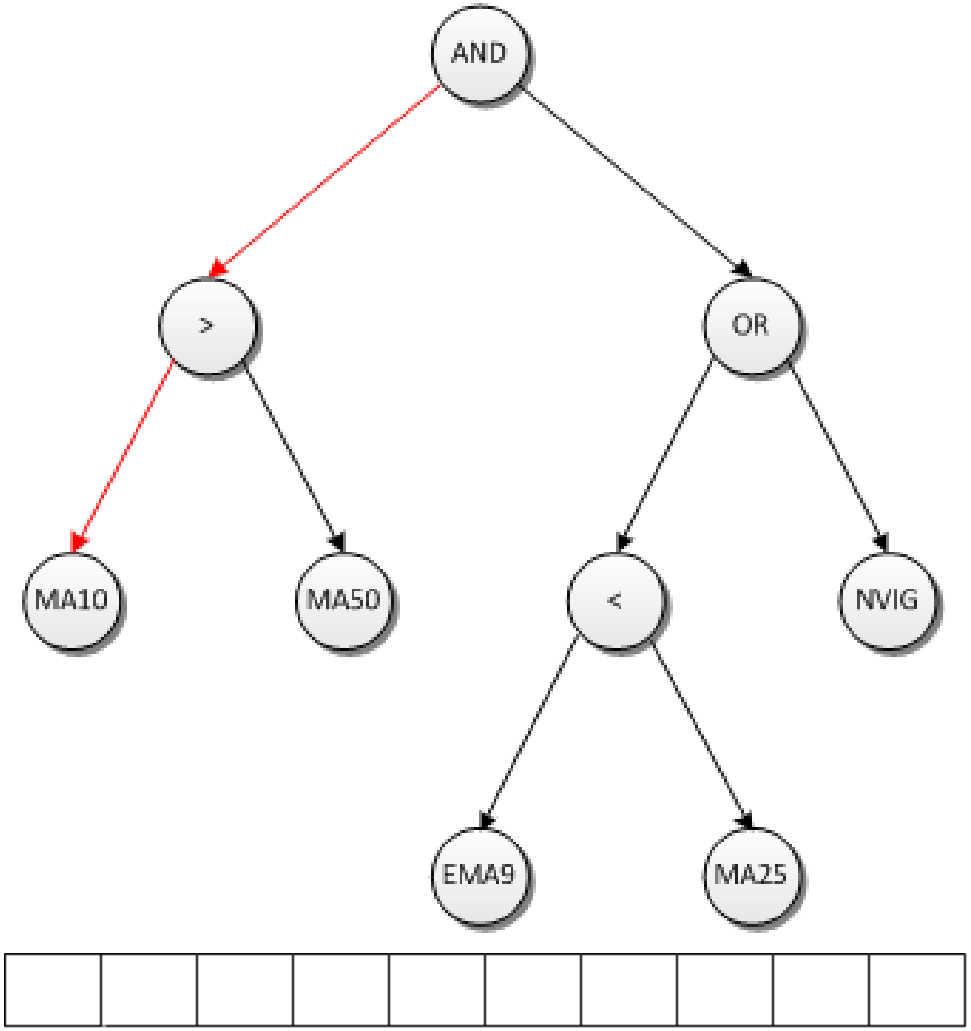
Example Conversion



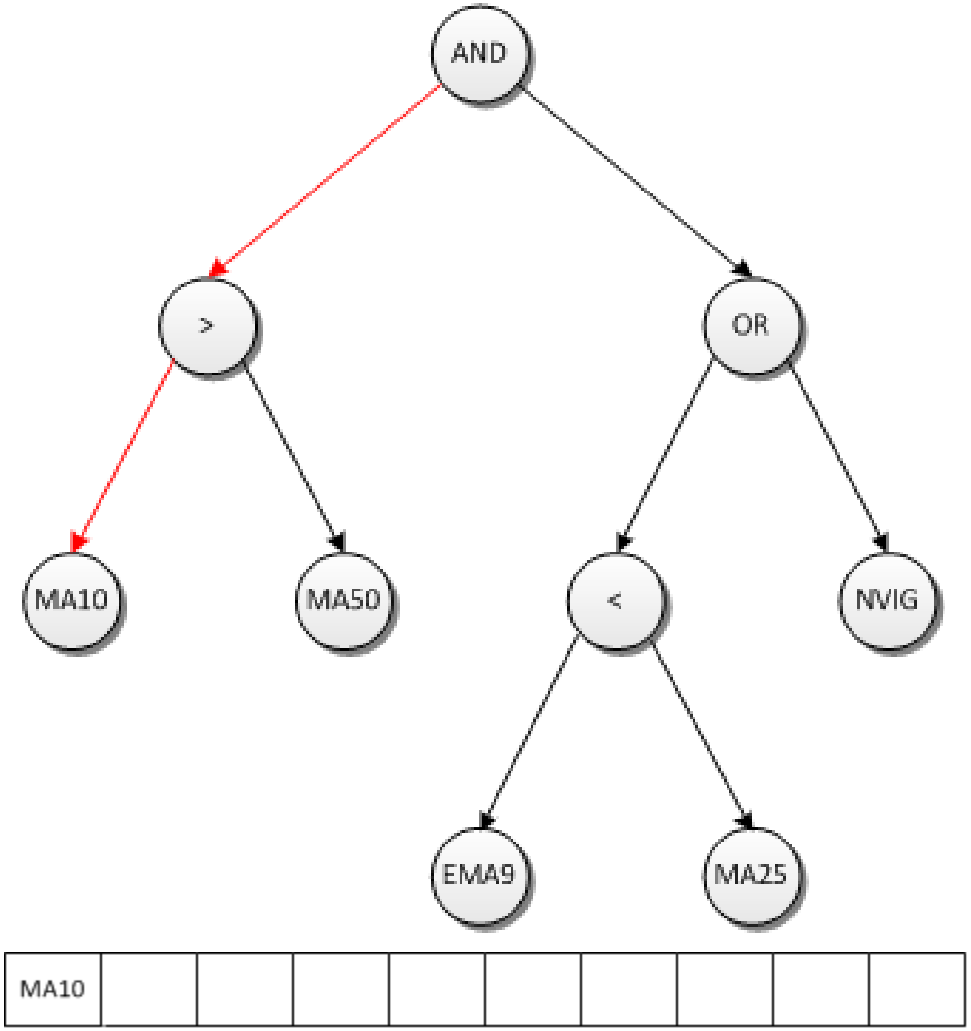
Example Conversion



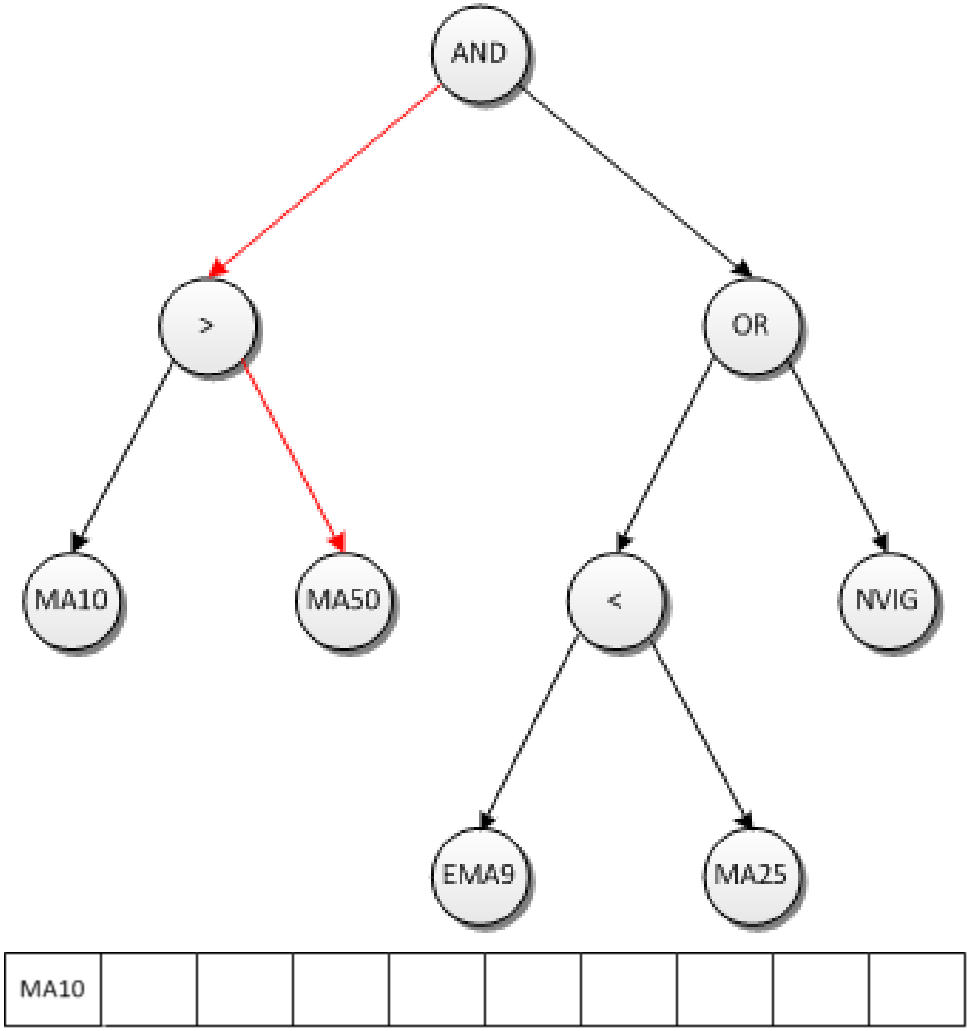
Example Conversion



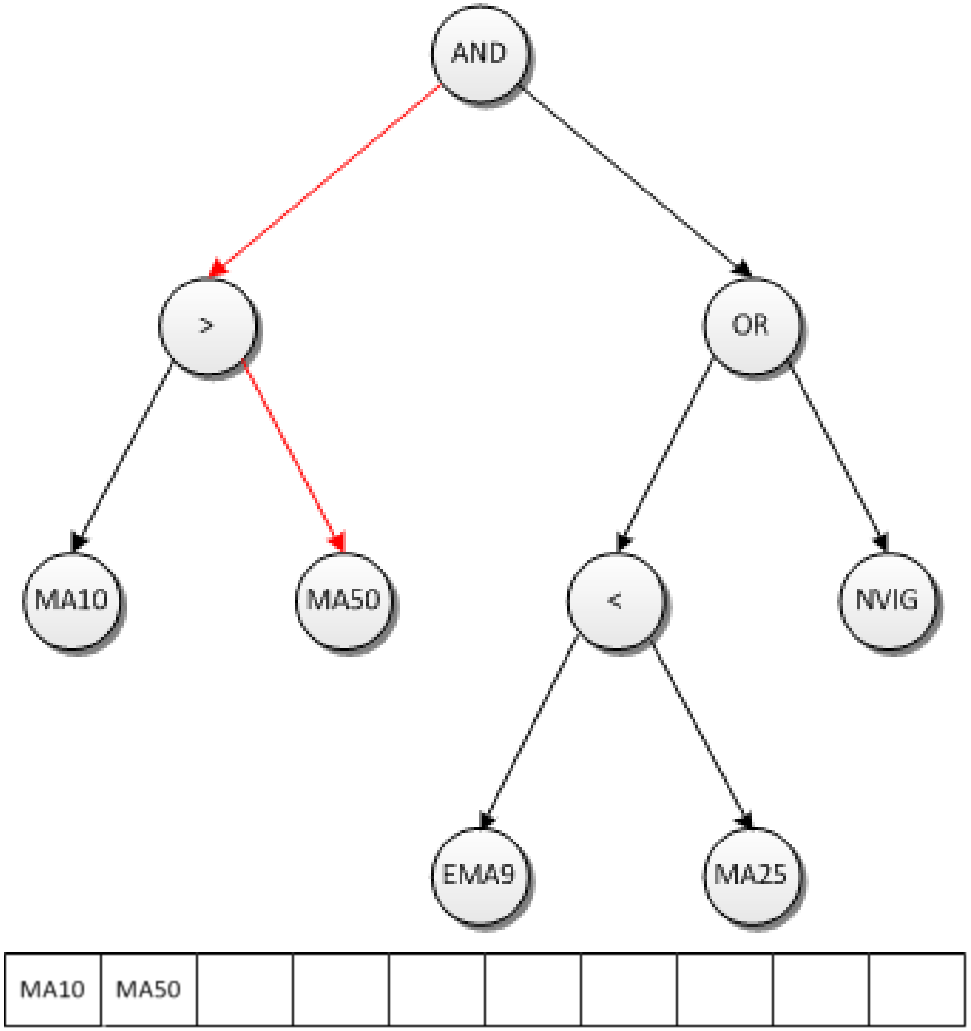
Example Conversion



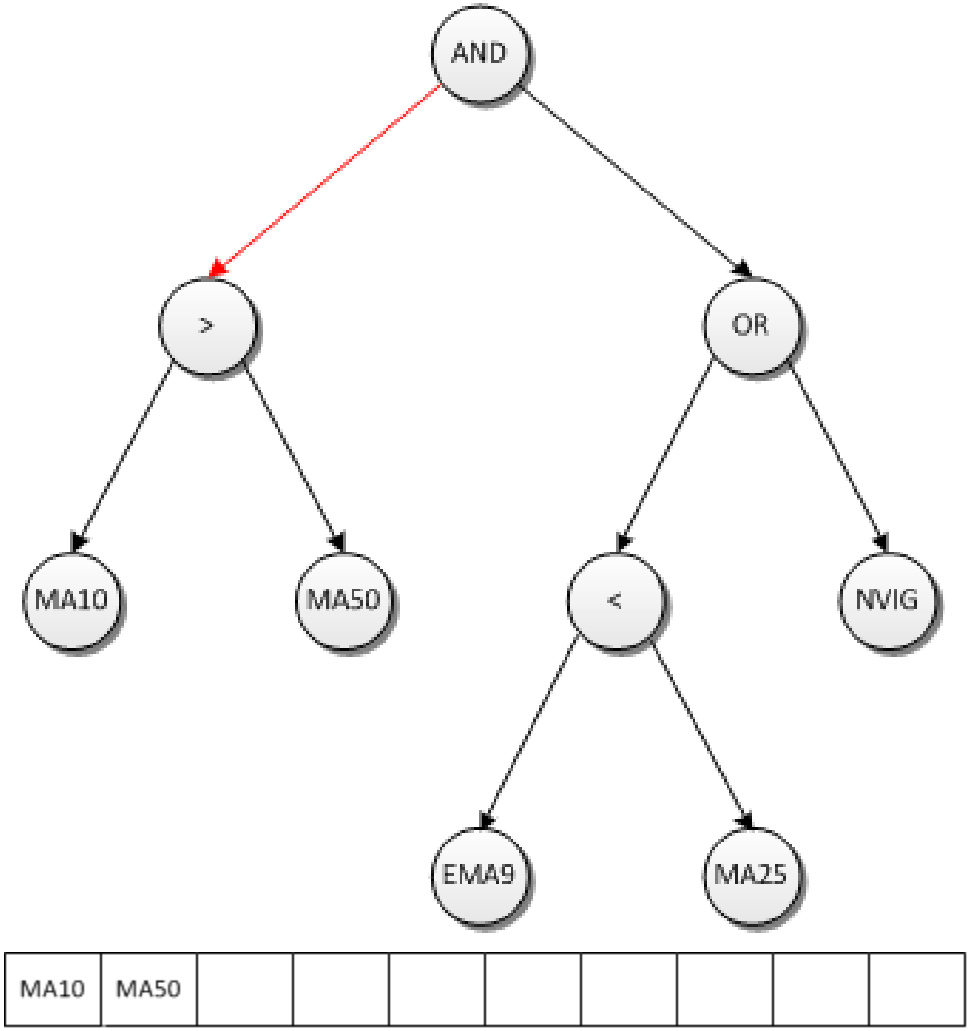
Example Conversion



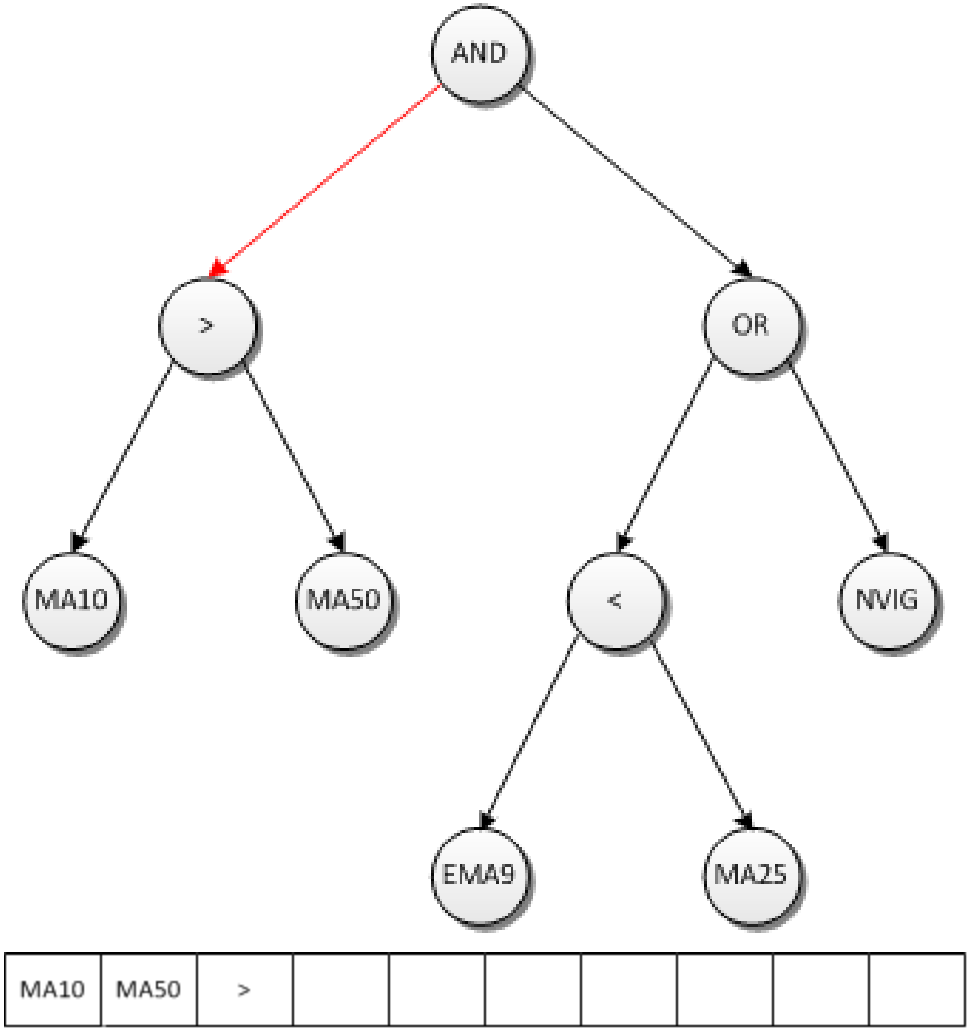
Example Conversion



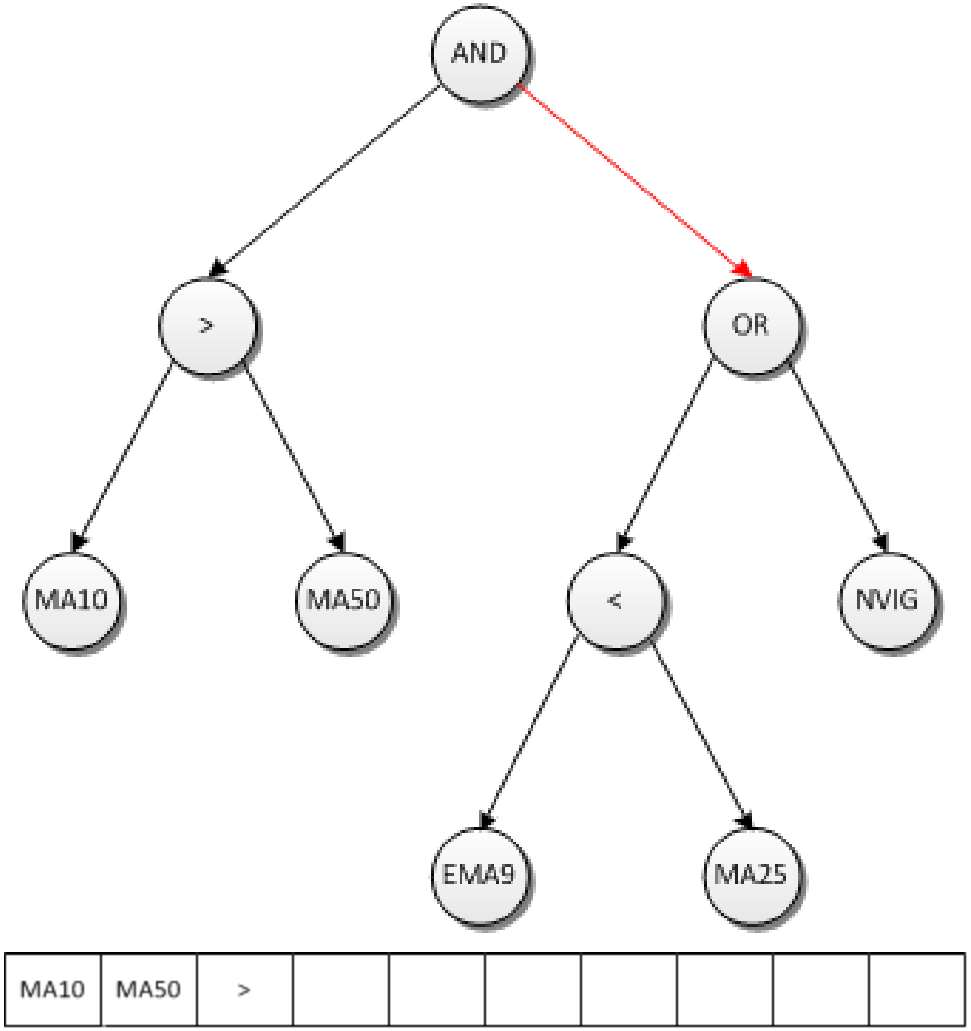
Example Conversion



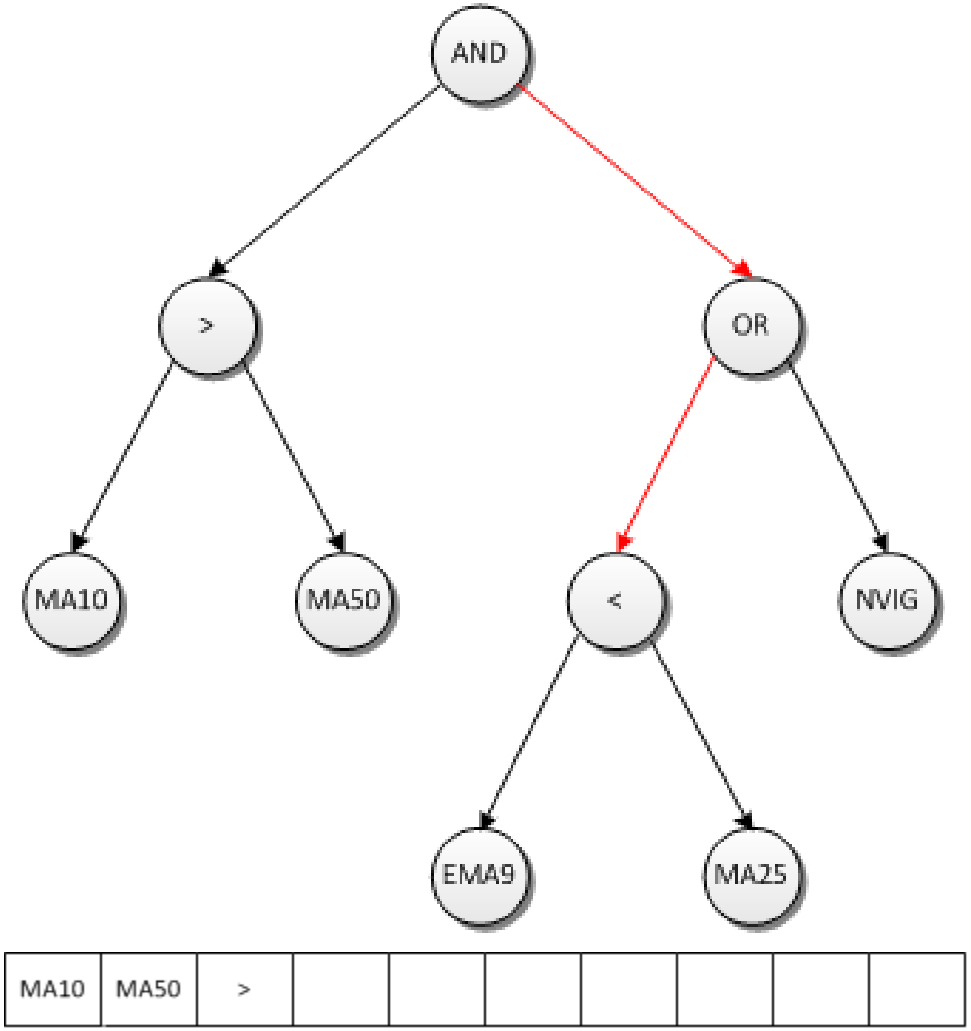
Example Conversion



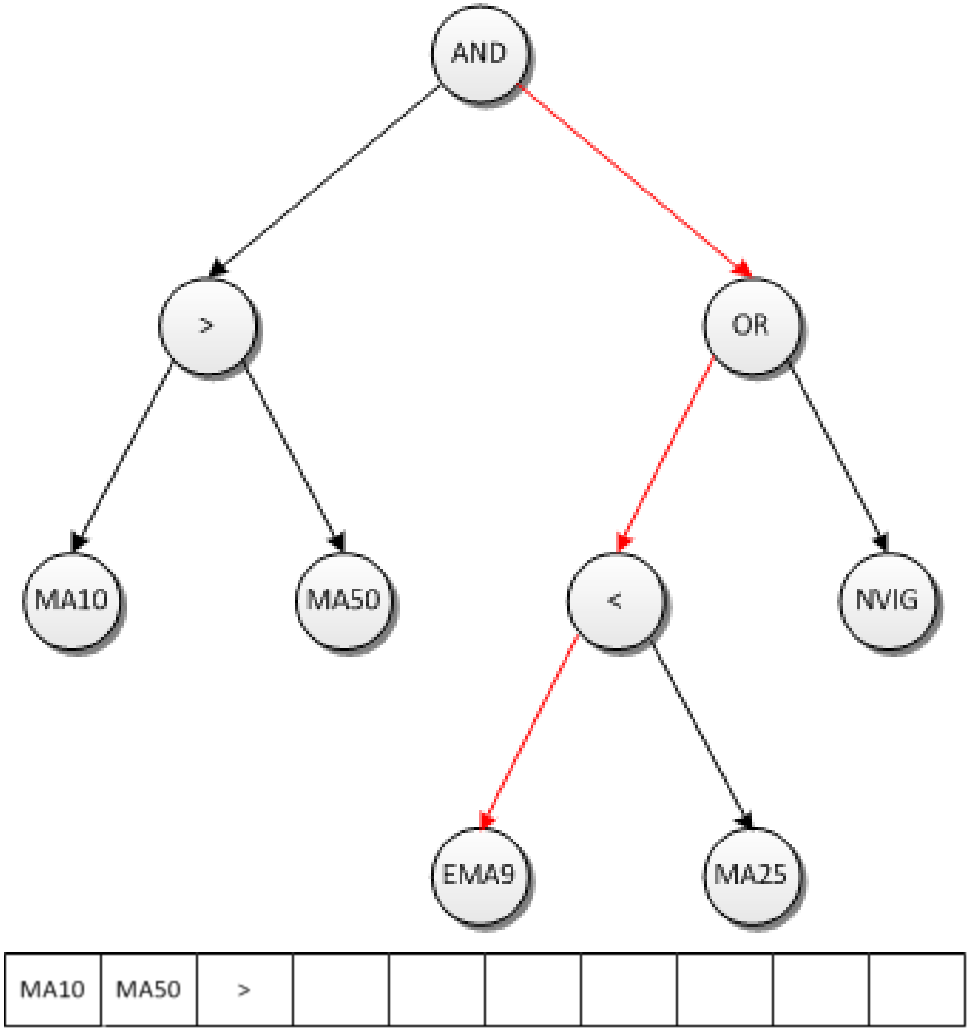
Example Conversion



Example Conversion

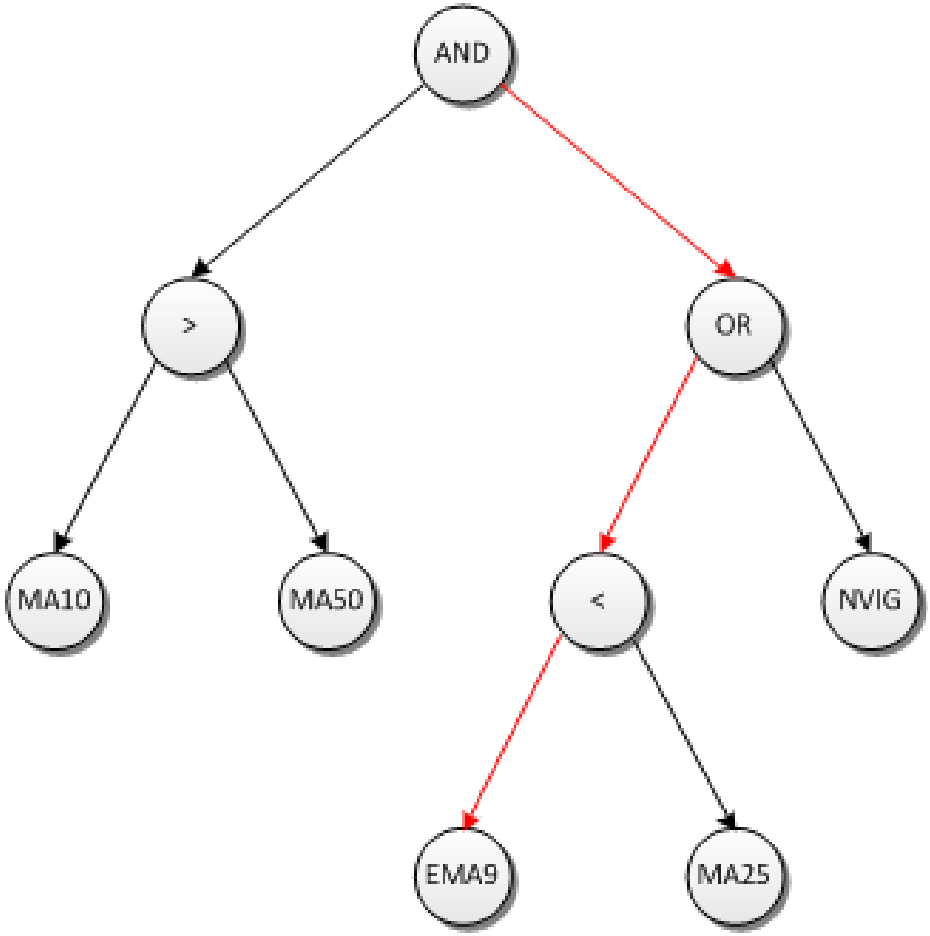


Example Conversion



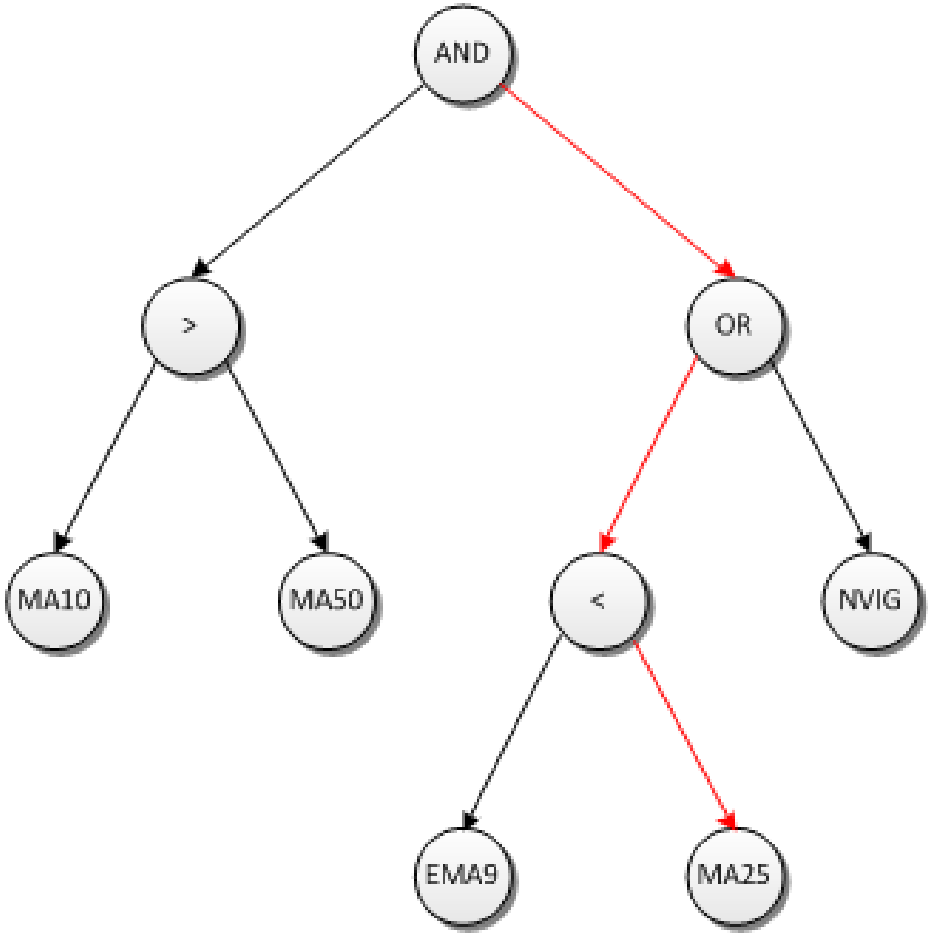
MA10	MA50	>							

Example Conversion



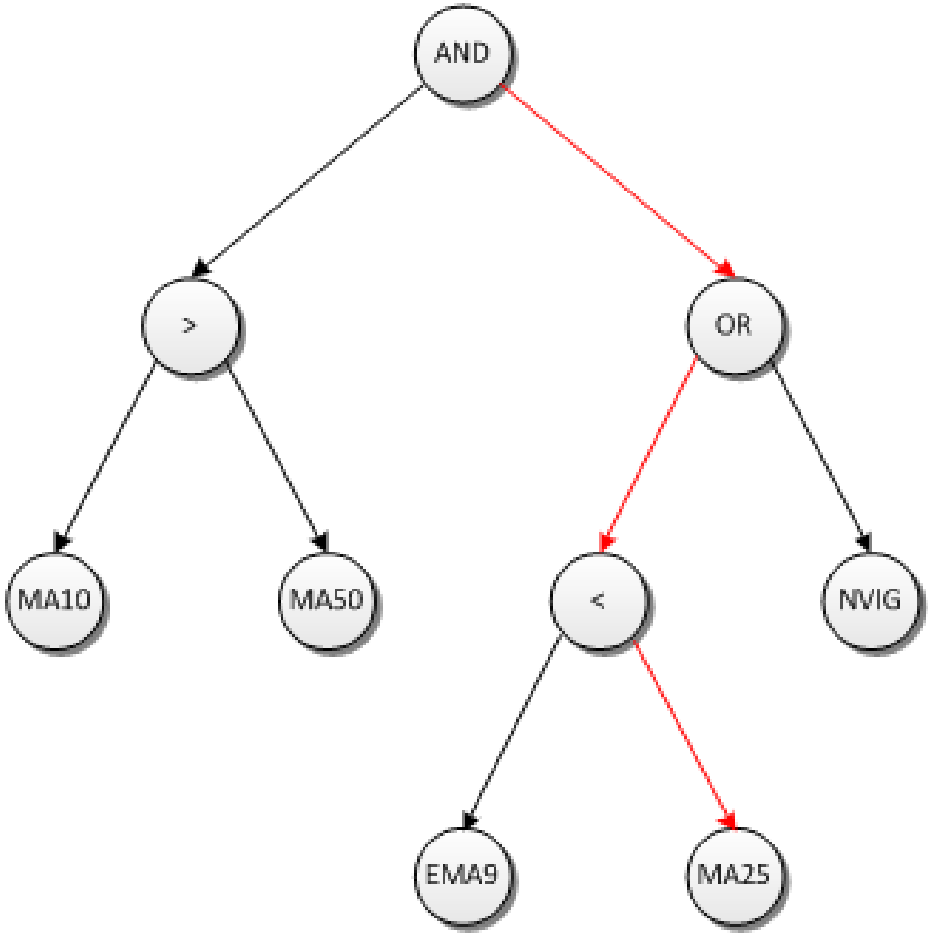
MA10	MA50	>	EMA9						

Example Conversion



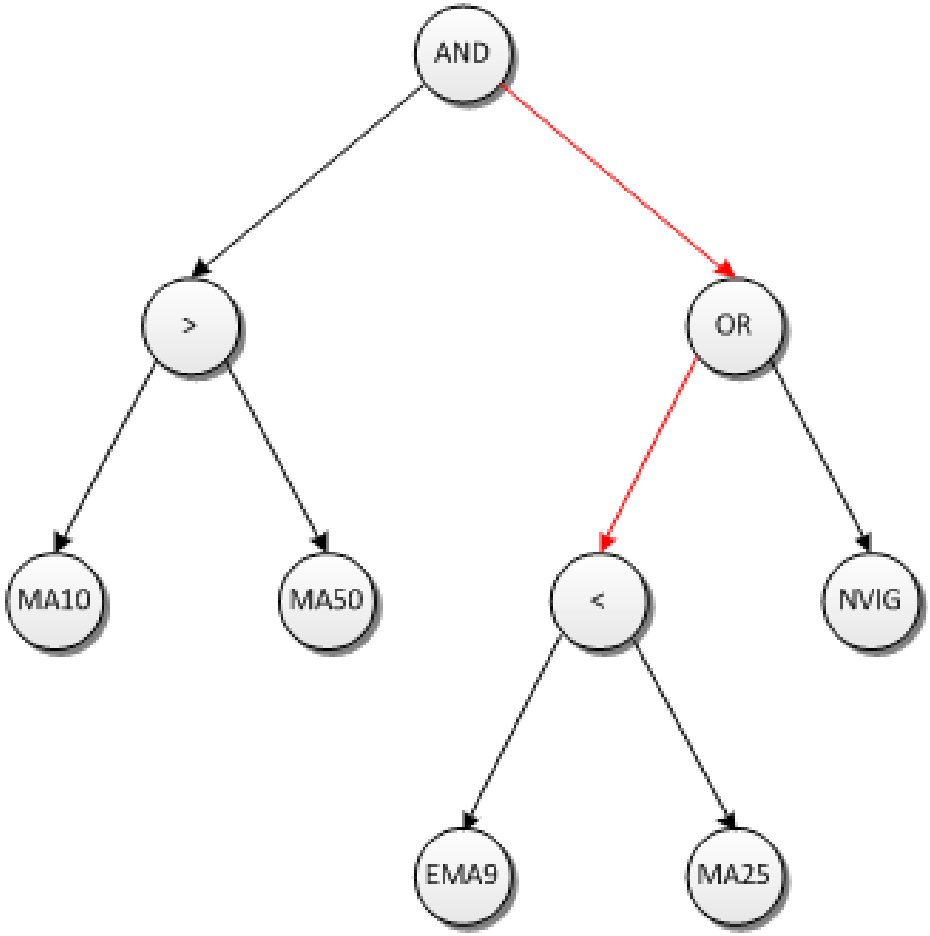
MA10	MA50	>	EMA9						

Example Conversion



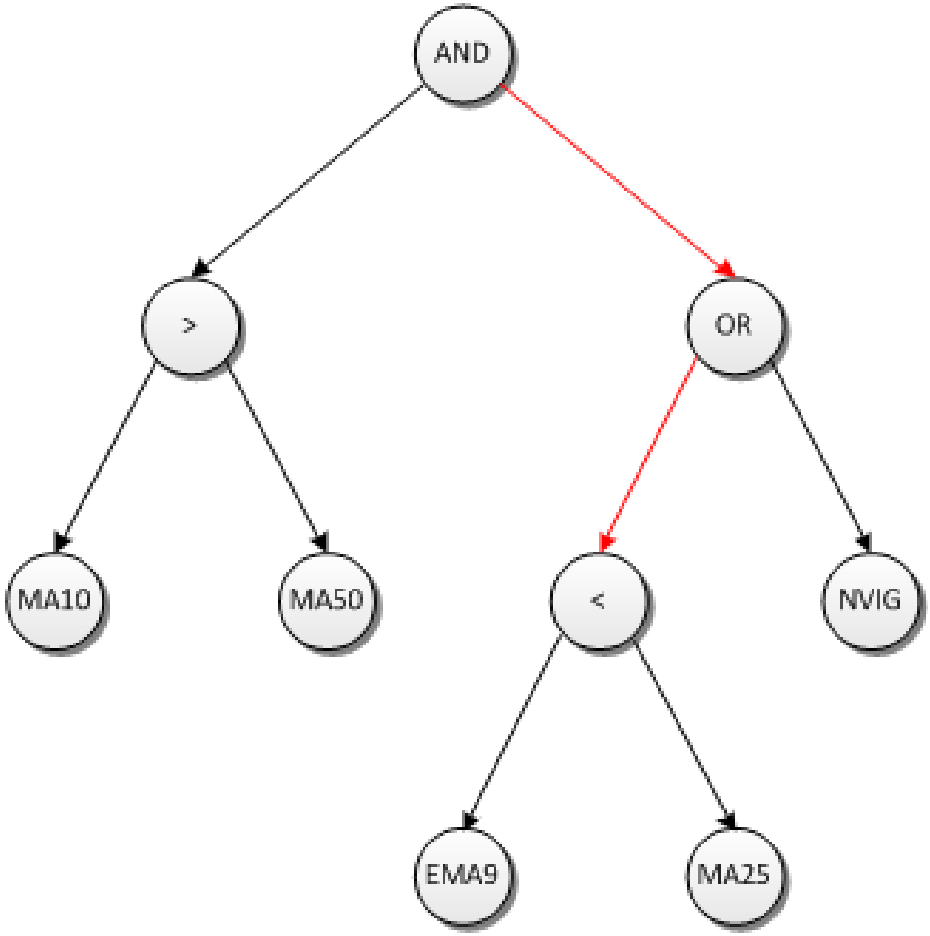
MA10	MA50	>	EMA9						

Example Conversion



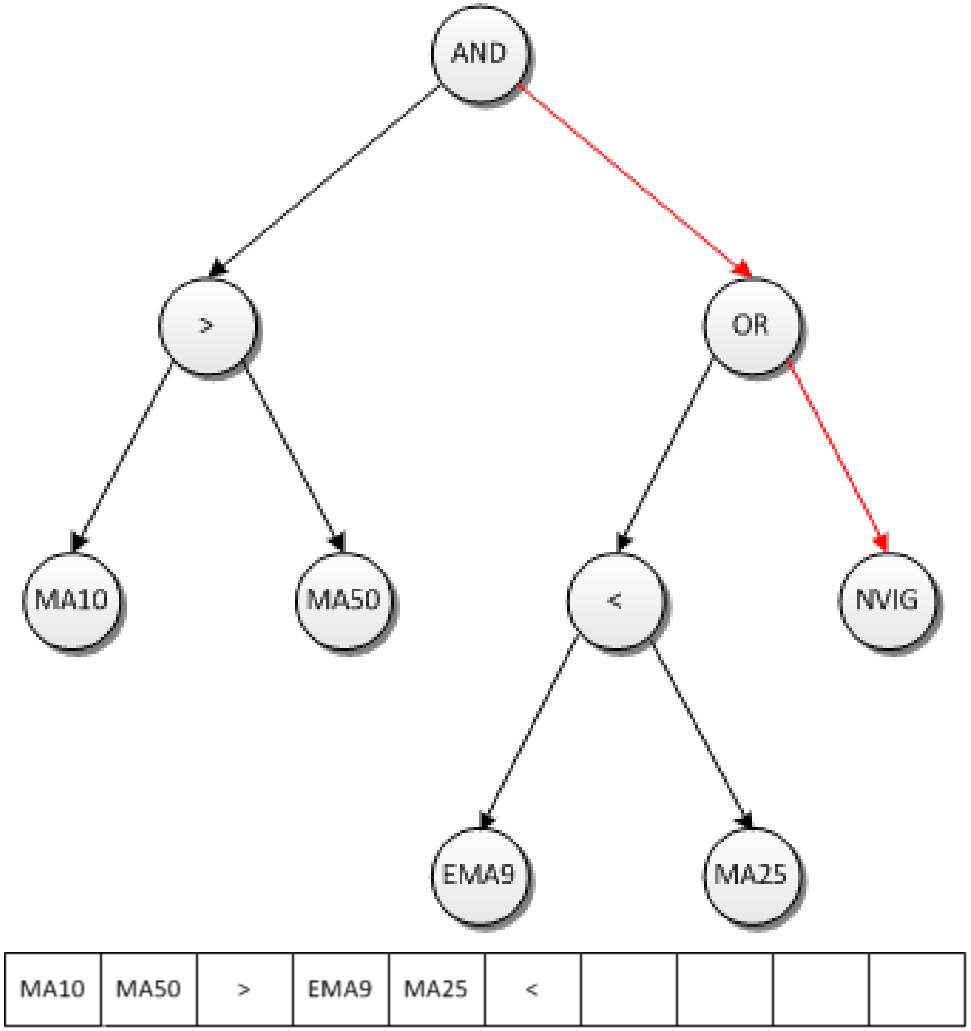
MA10	MA50	>	EMA9	MA25					

Example Conversion



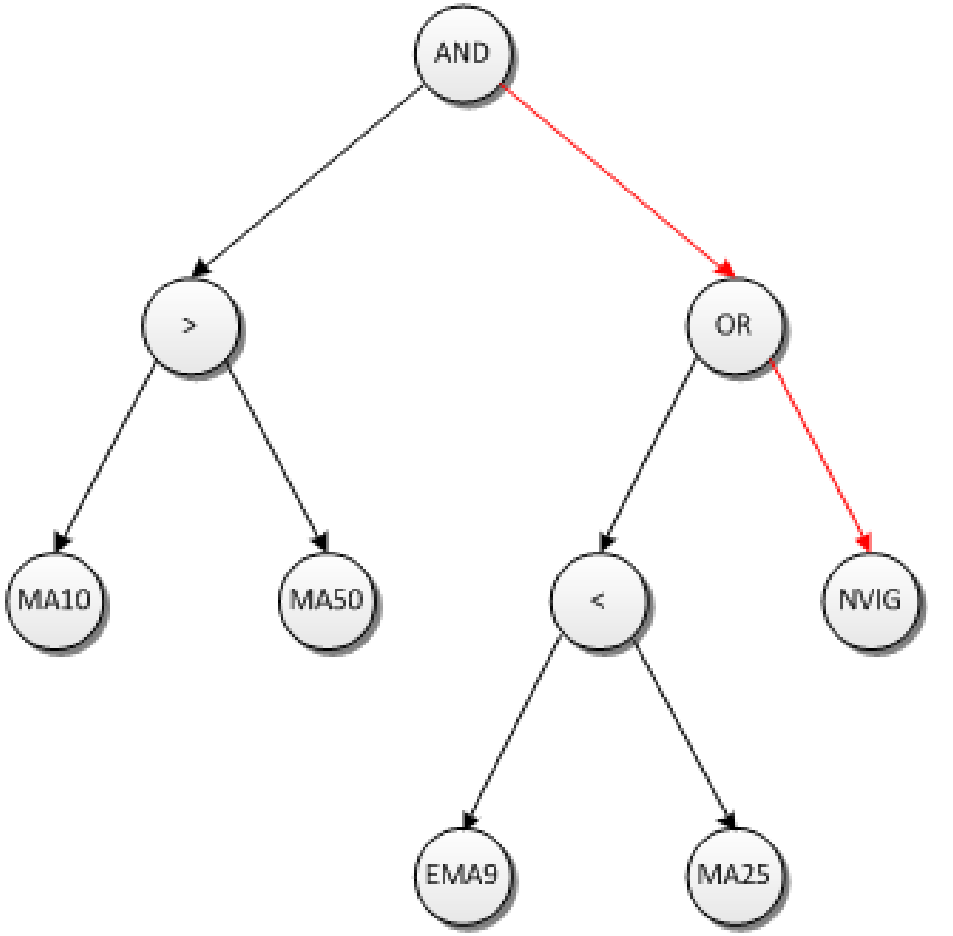
MA10	MA50	>	EMA9	MA25	<				

Example Conversion



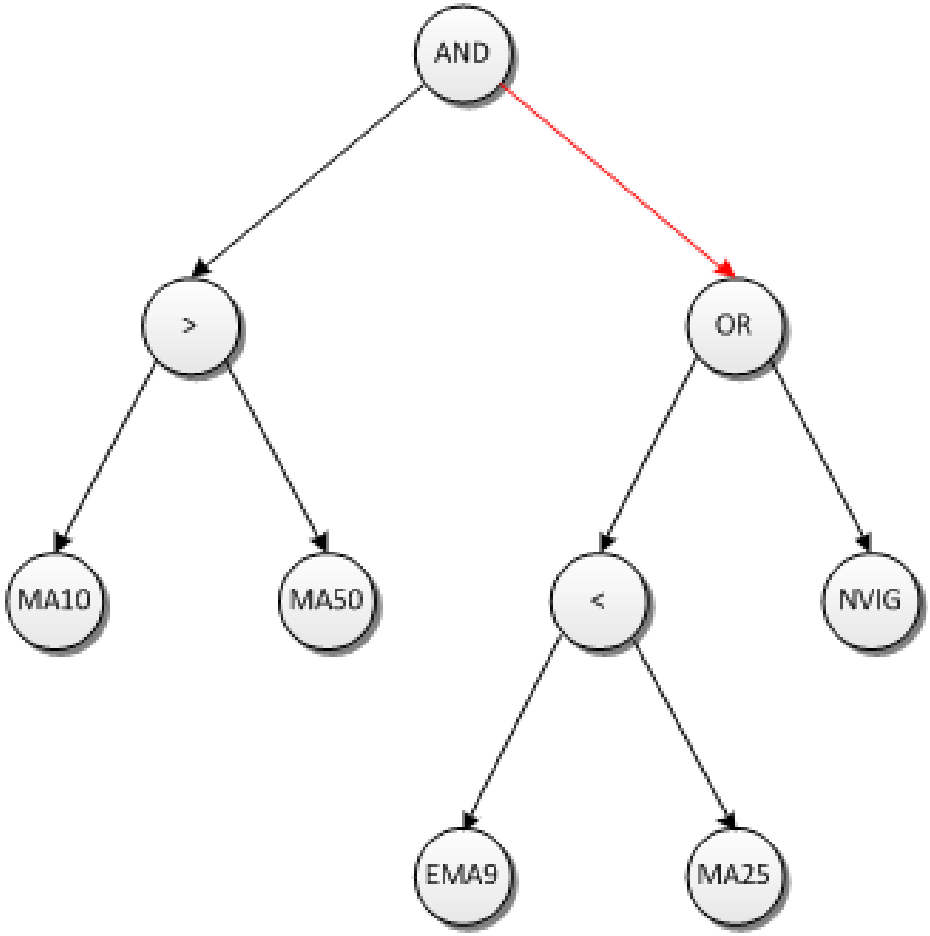
MA10	MA50	>	EMA9	MA25	<				

Example Conversion



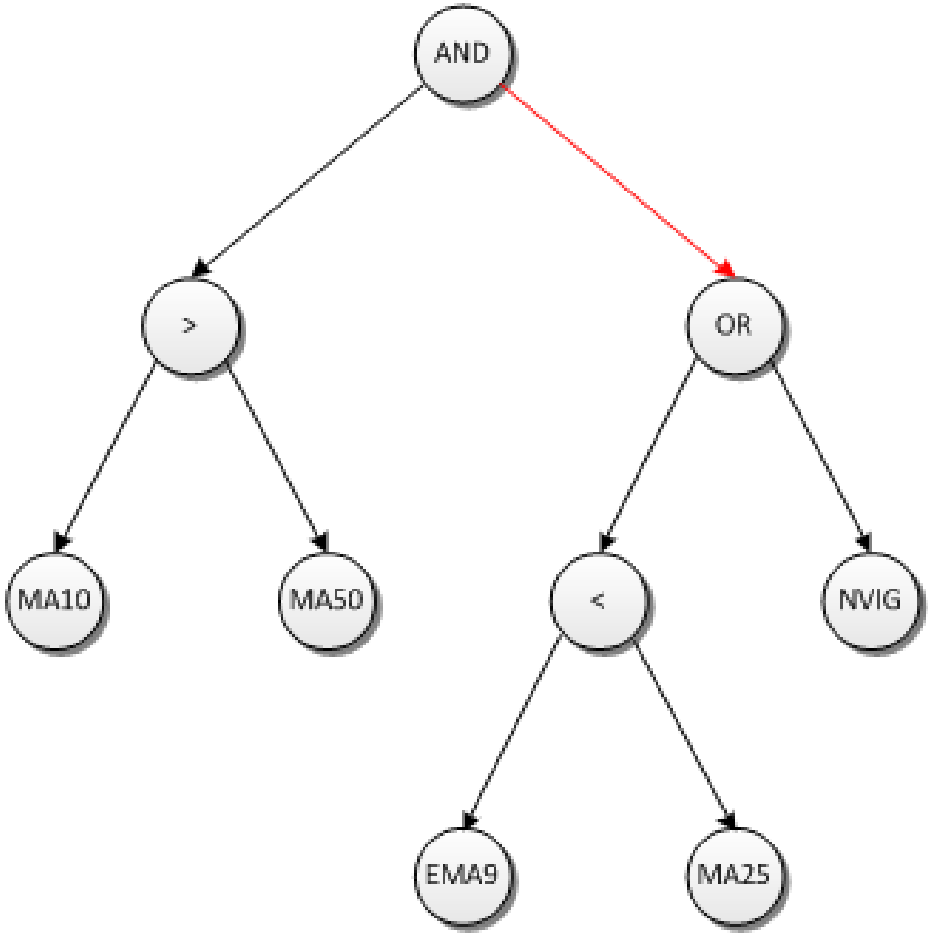
MA10	MA50	>	EMA9	MA25	<	NVIG			

Example Conversion



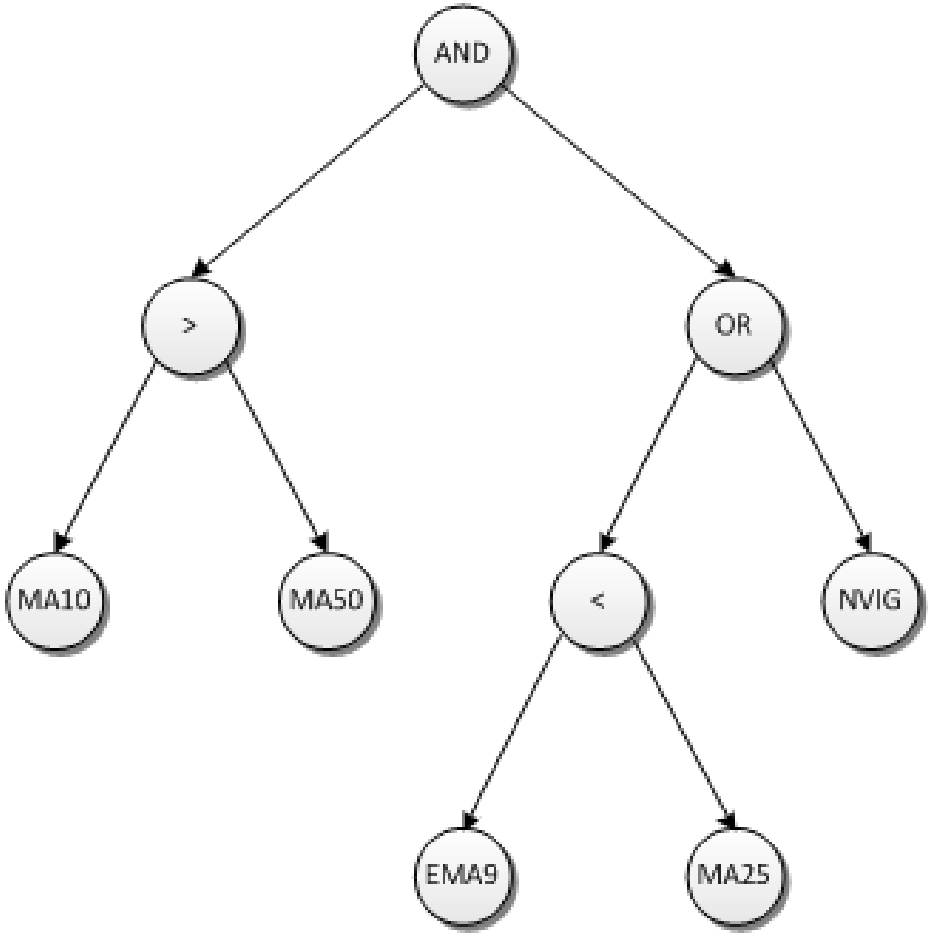
MA10	MA50	>	EMA9	MA25	<	NVIG			

Example Conversion



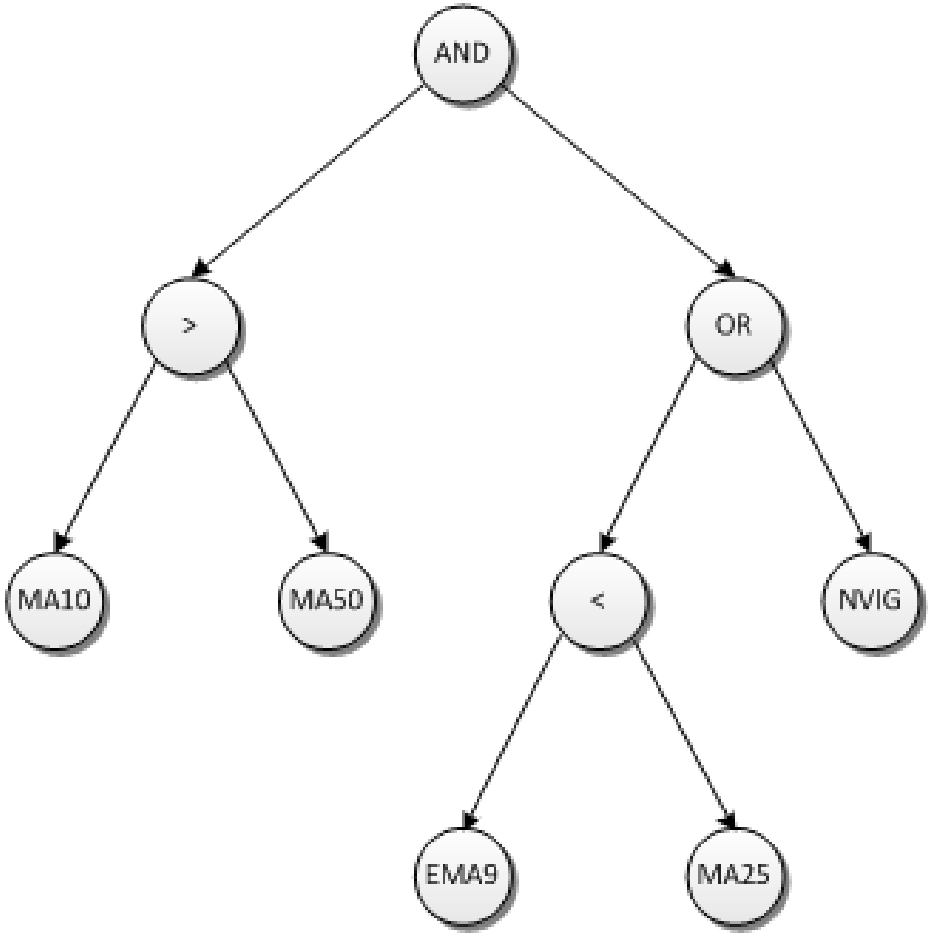
MA10	MA50	>	EMA9	MA25	<	NVIG	OR		

Example Conversion



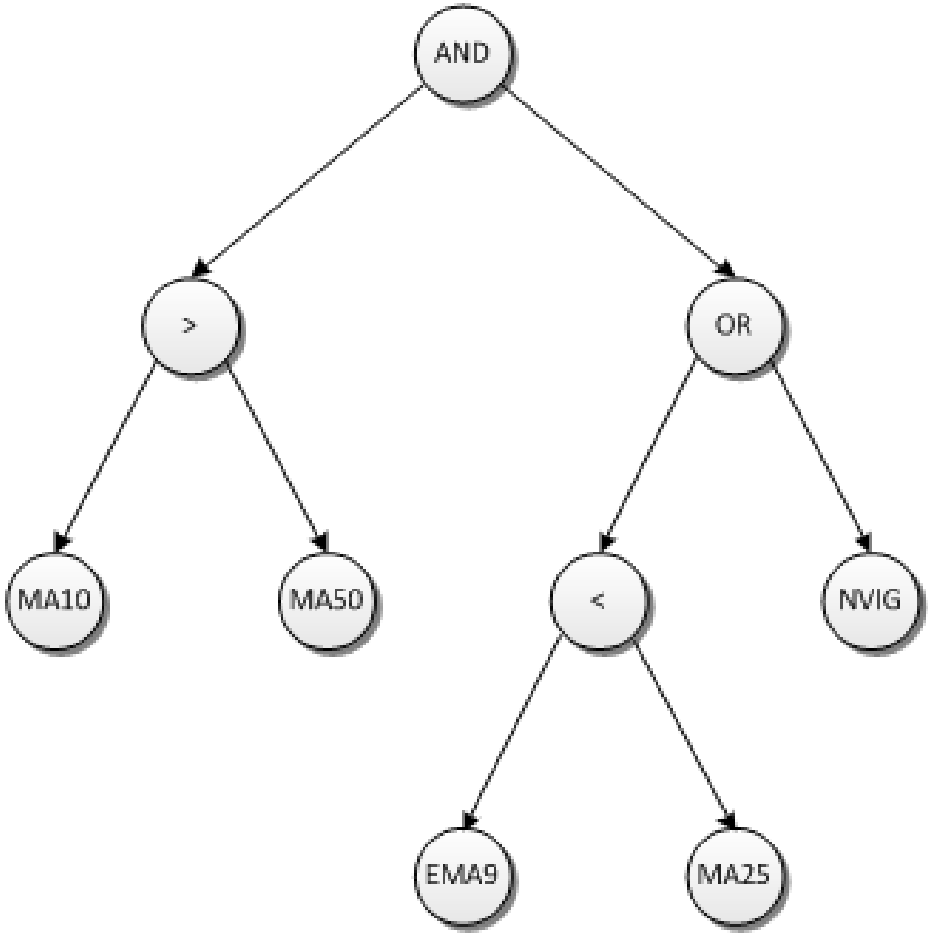
MA10	MA50	>	EMA9	MA25	<	NVIG	OR		

Example Conversion



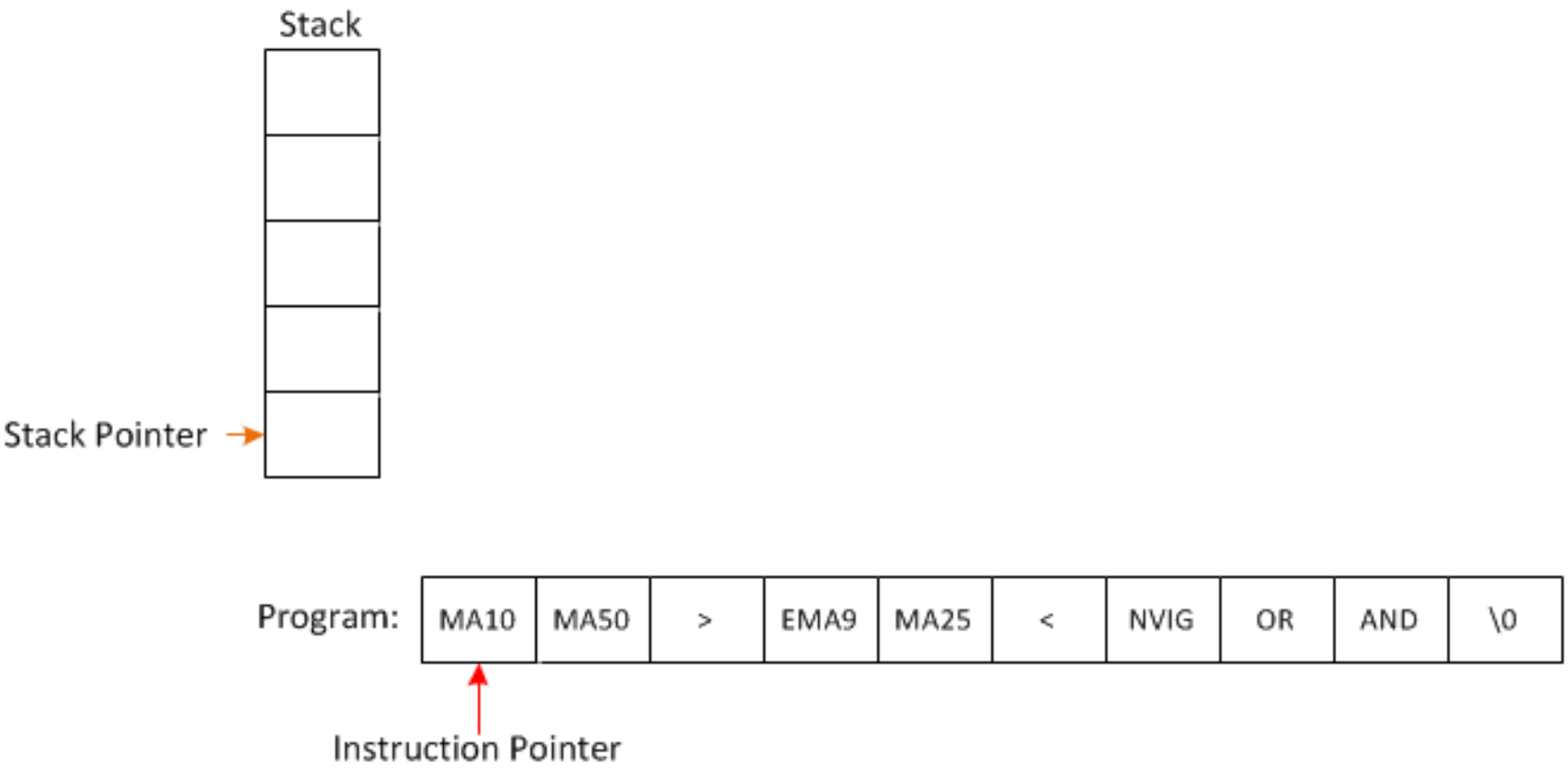
MA10	MA50	>	EMA9	MA25	<	NVIG	OR	AND	

Example Conversion

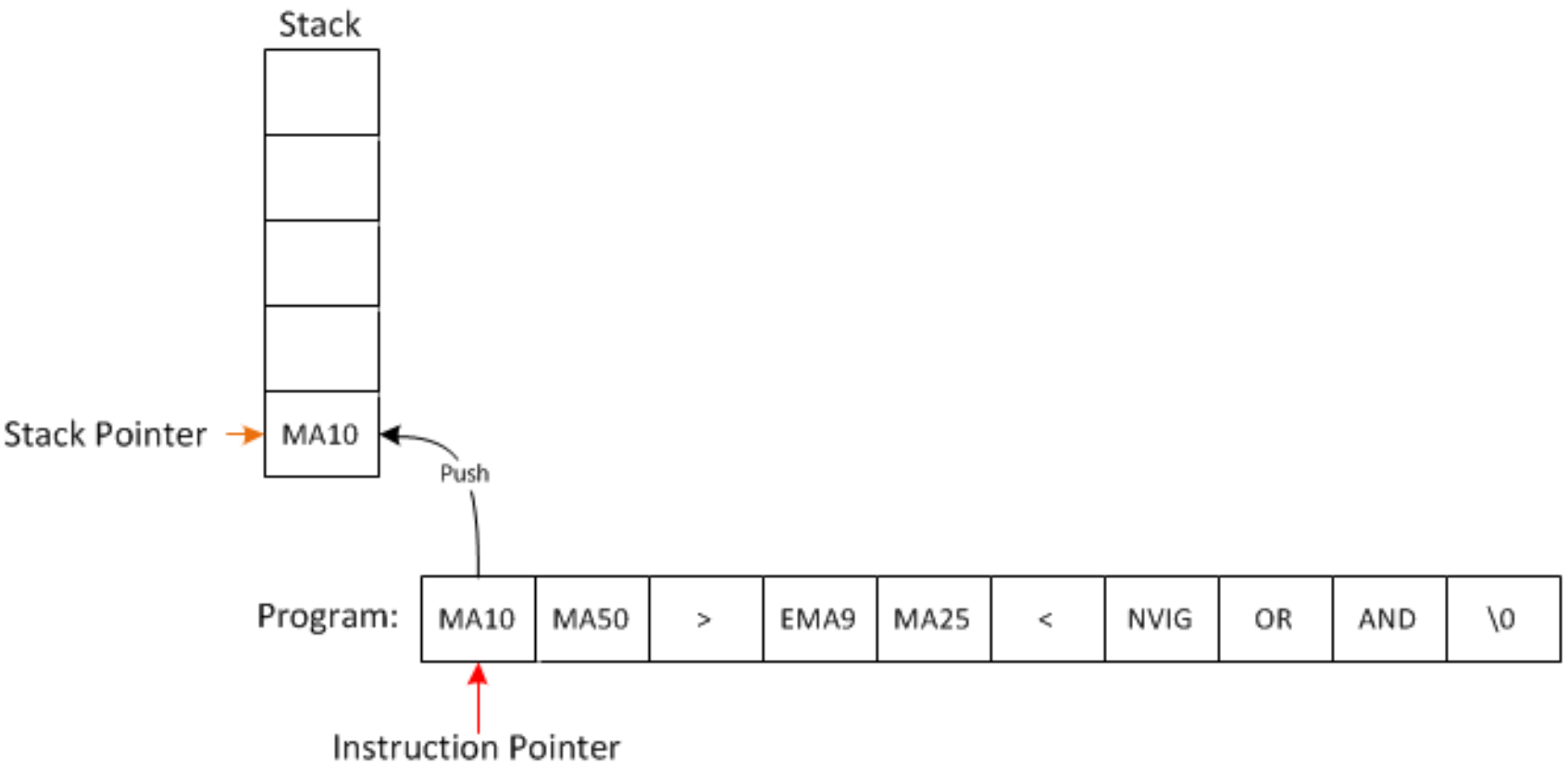


MA10	MA50	>	EMA9	MA25	<	NVIG	OR	AND	\0

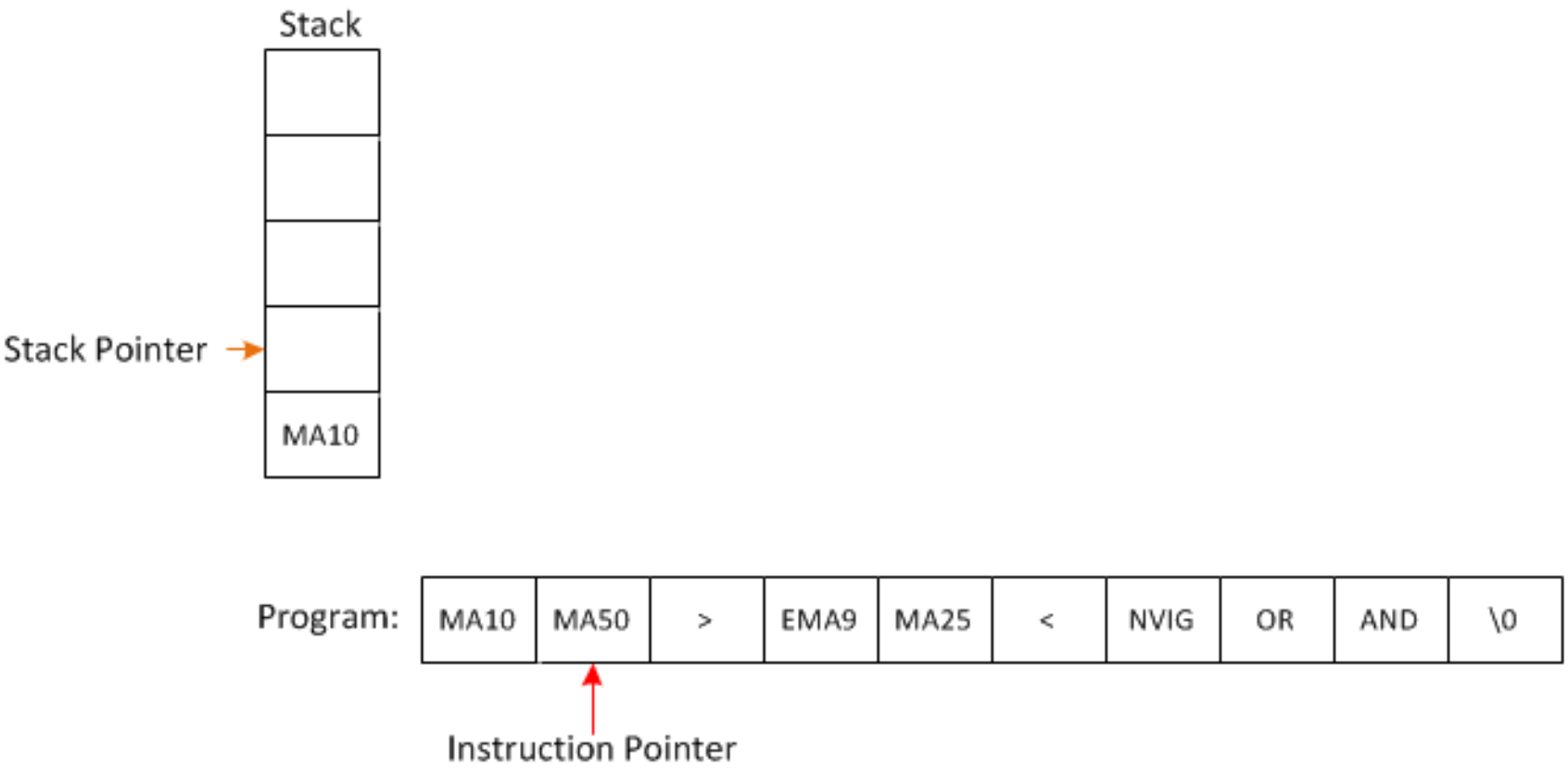
Stack-based Interpreter



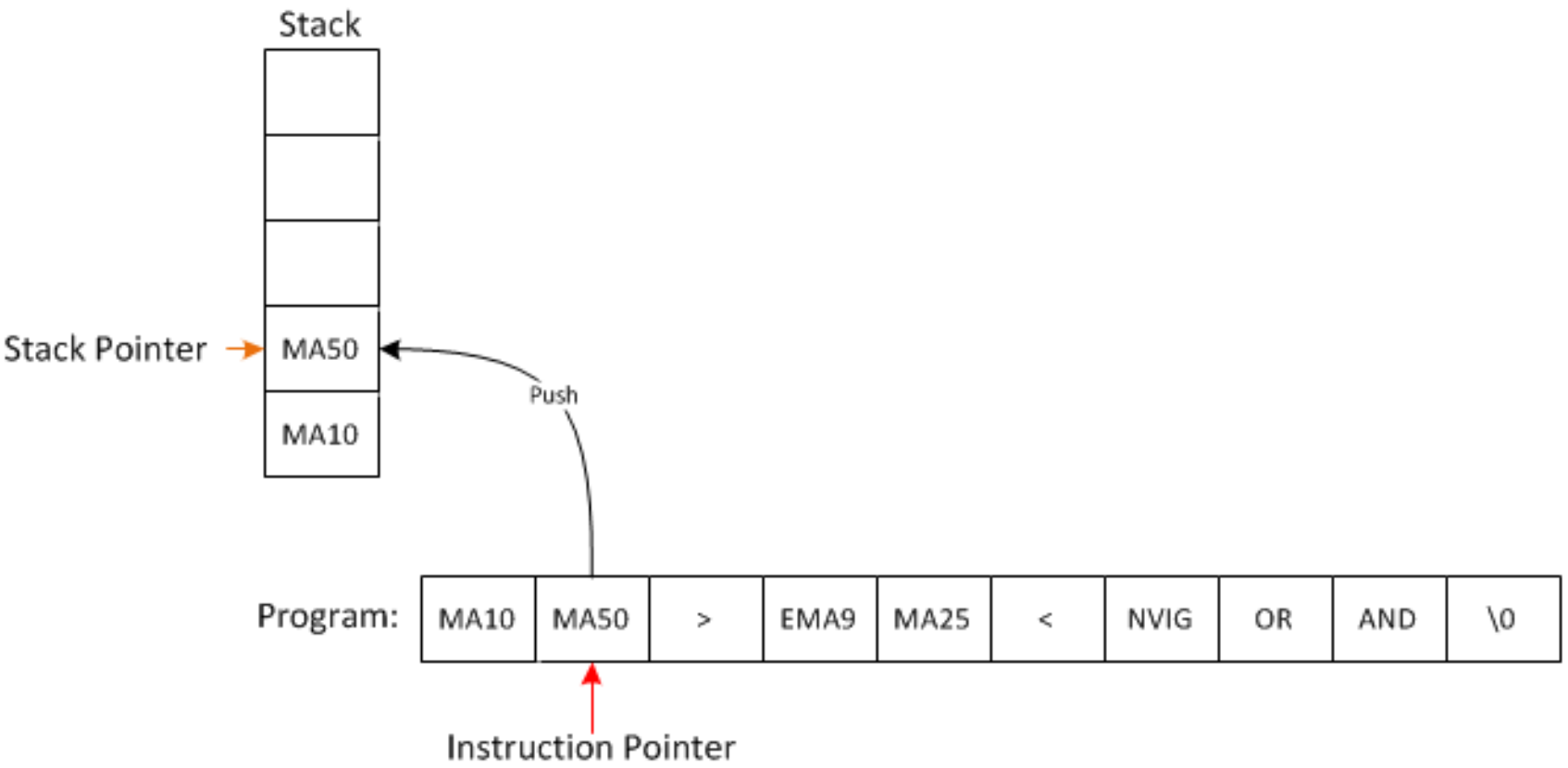
Stack-based Interpreter



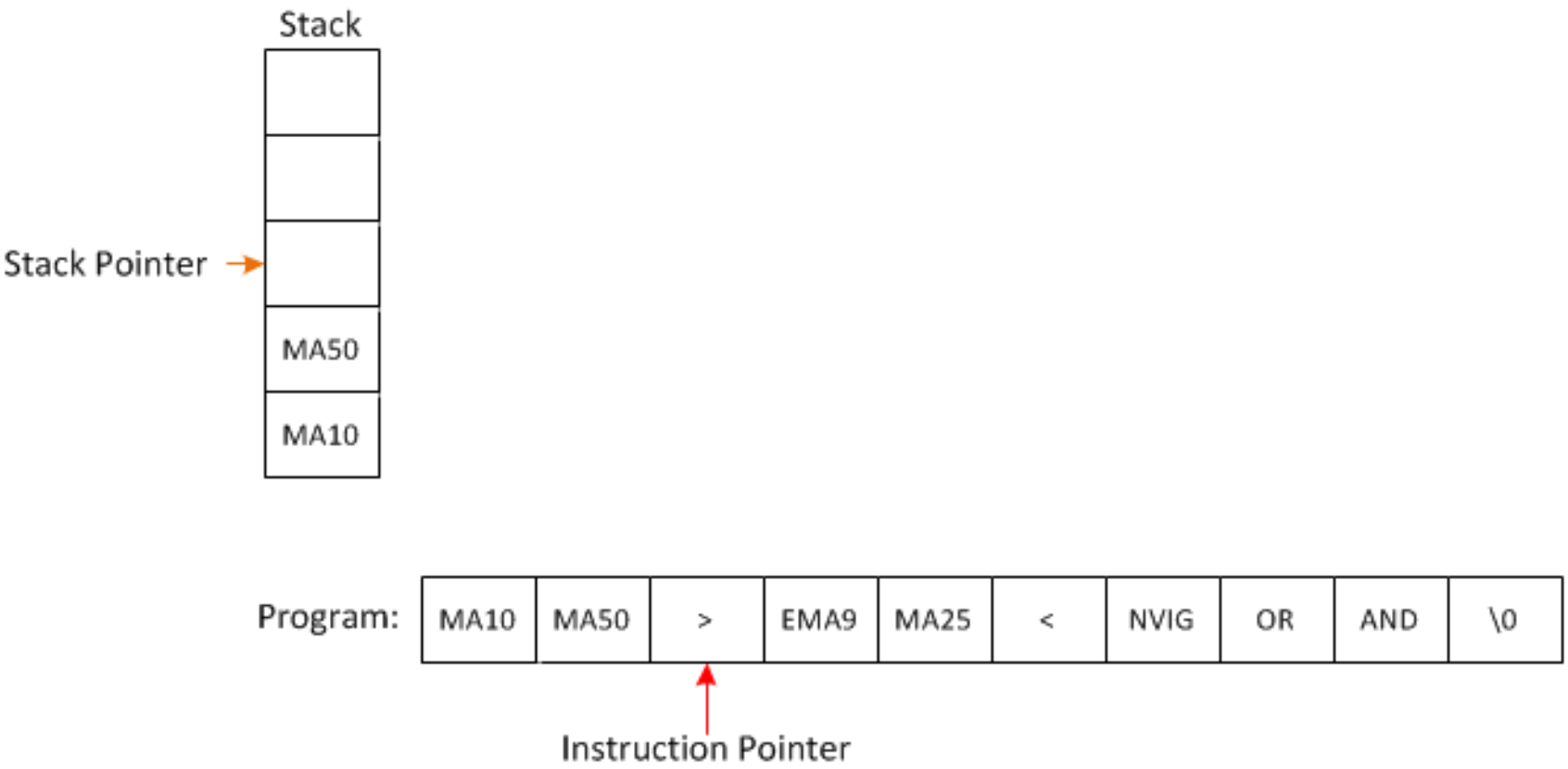
Stack-based Interpreter



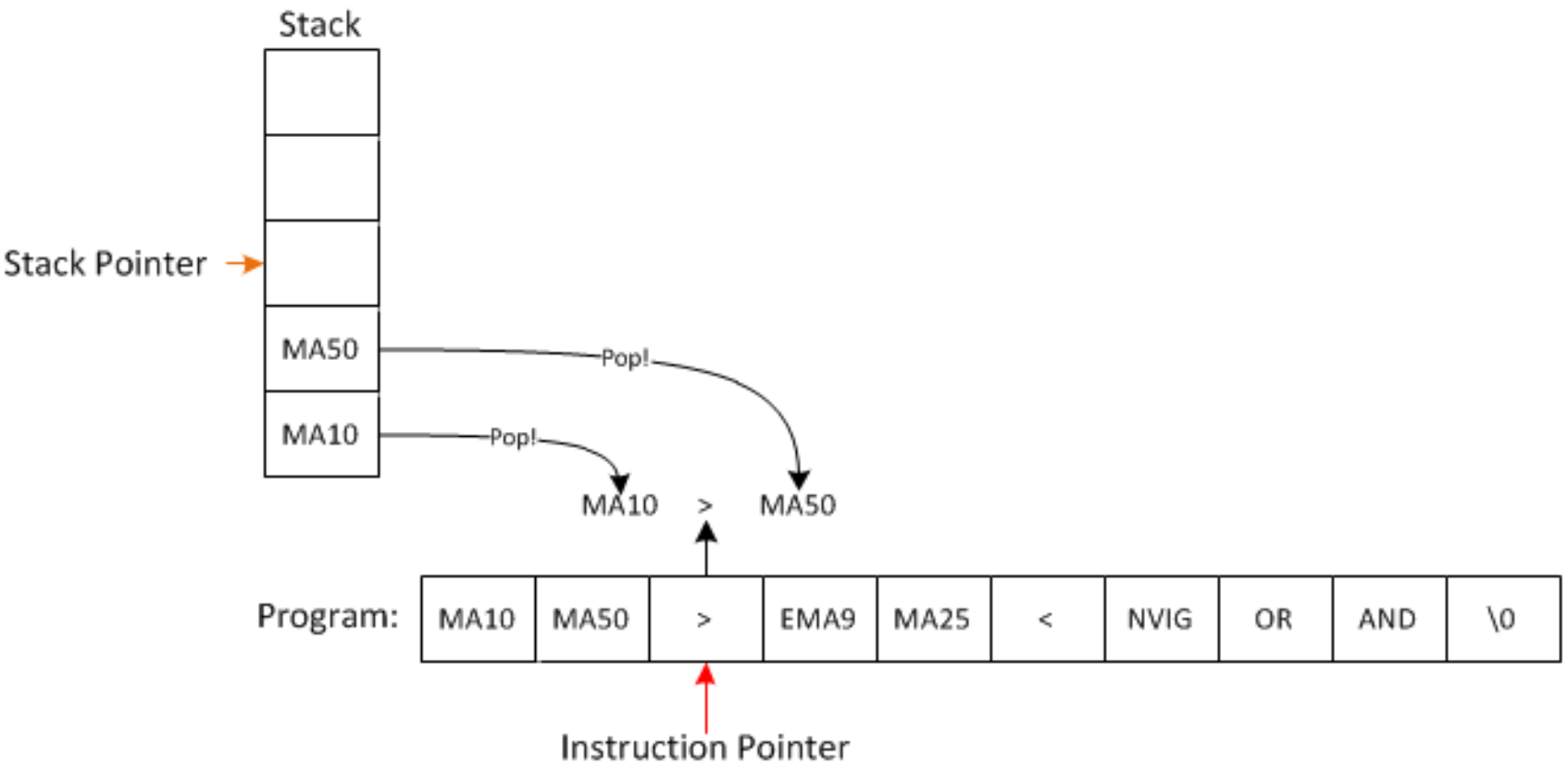
Stack-based Interpreter



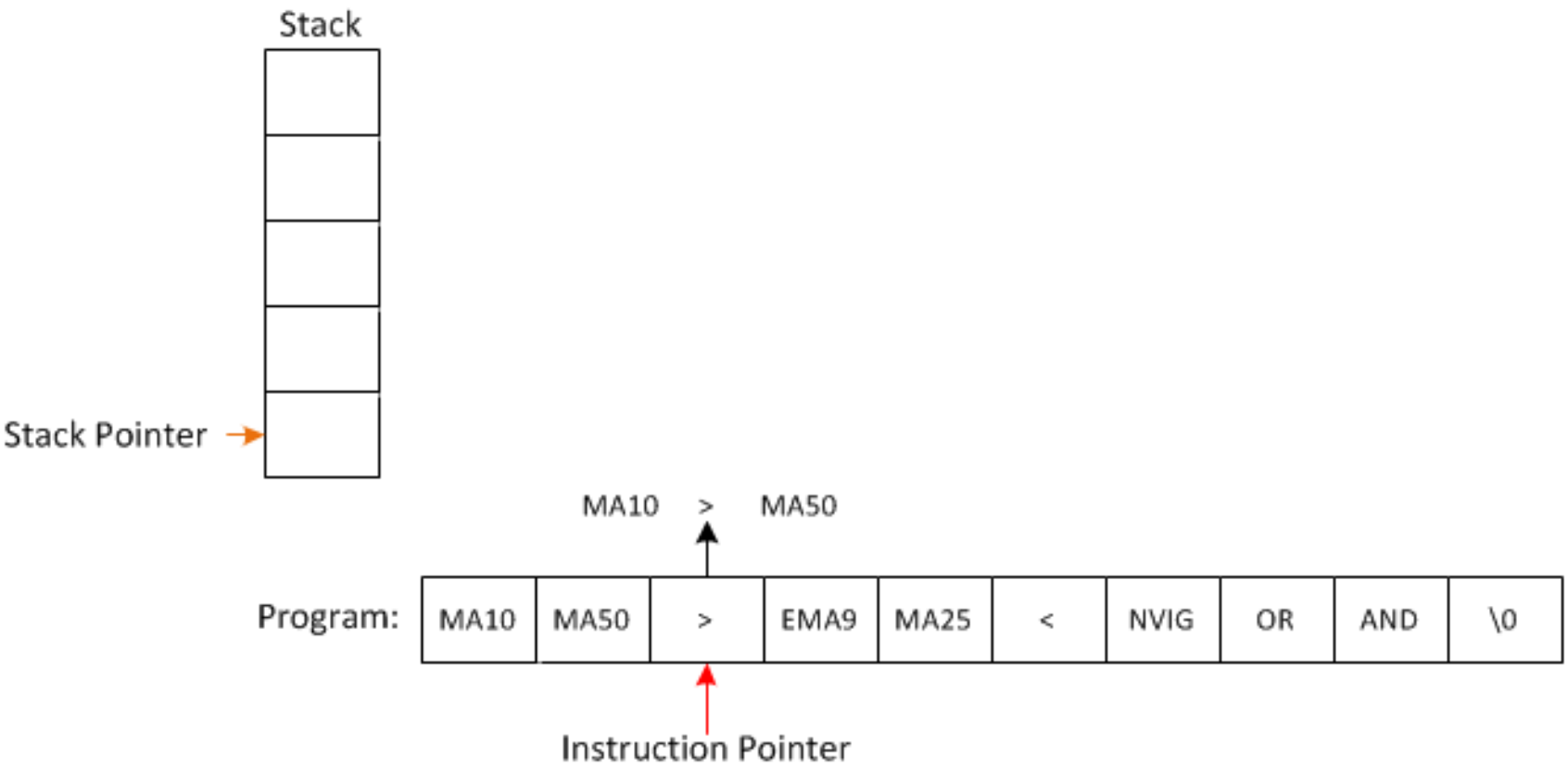
Stack-based Interpreter



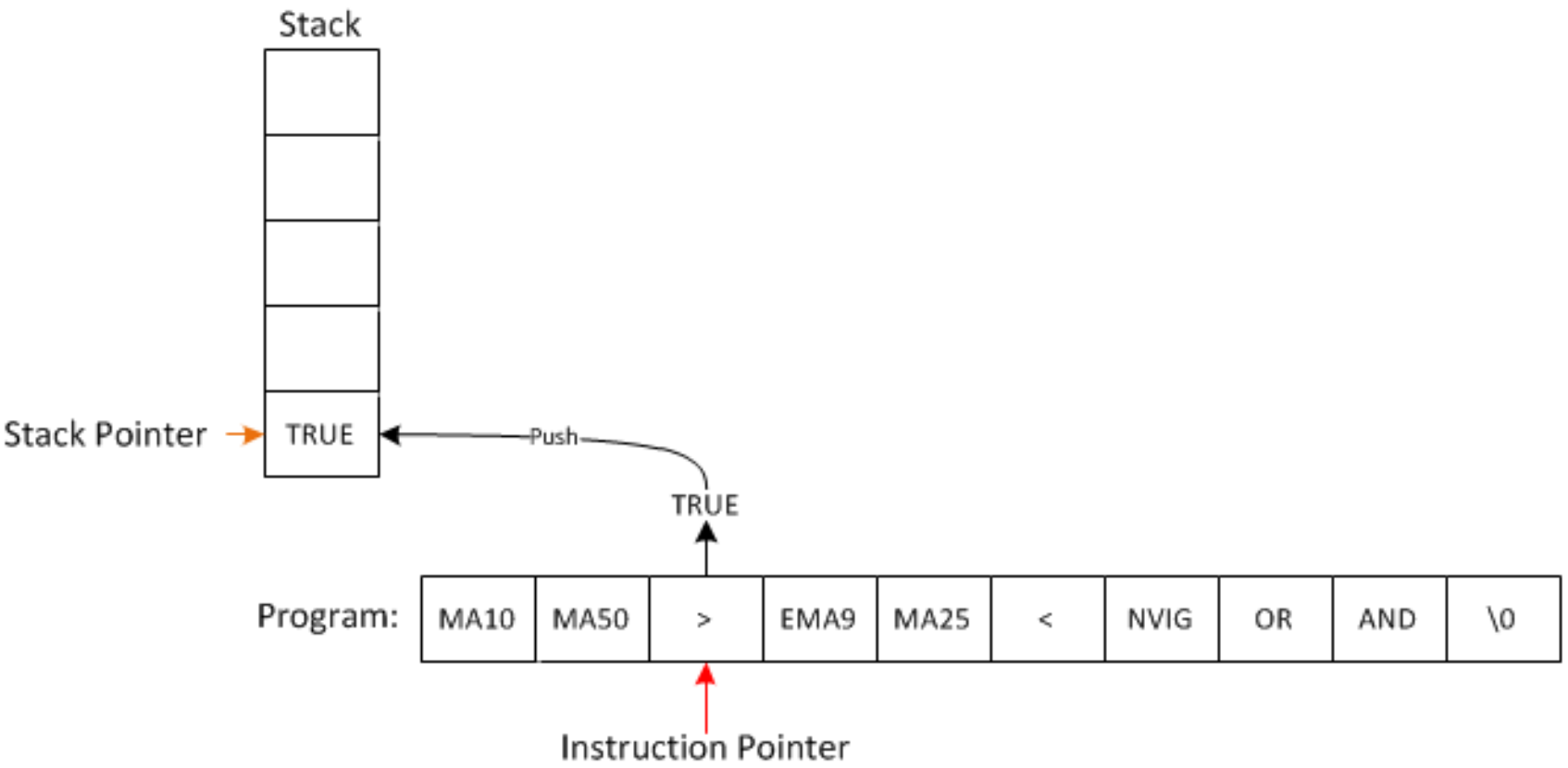
Stack-based Interpreter



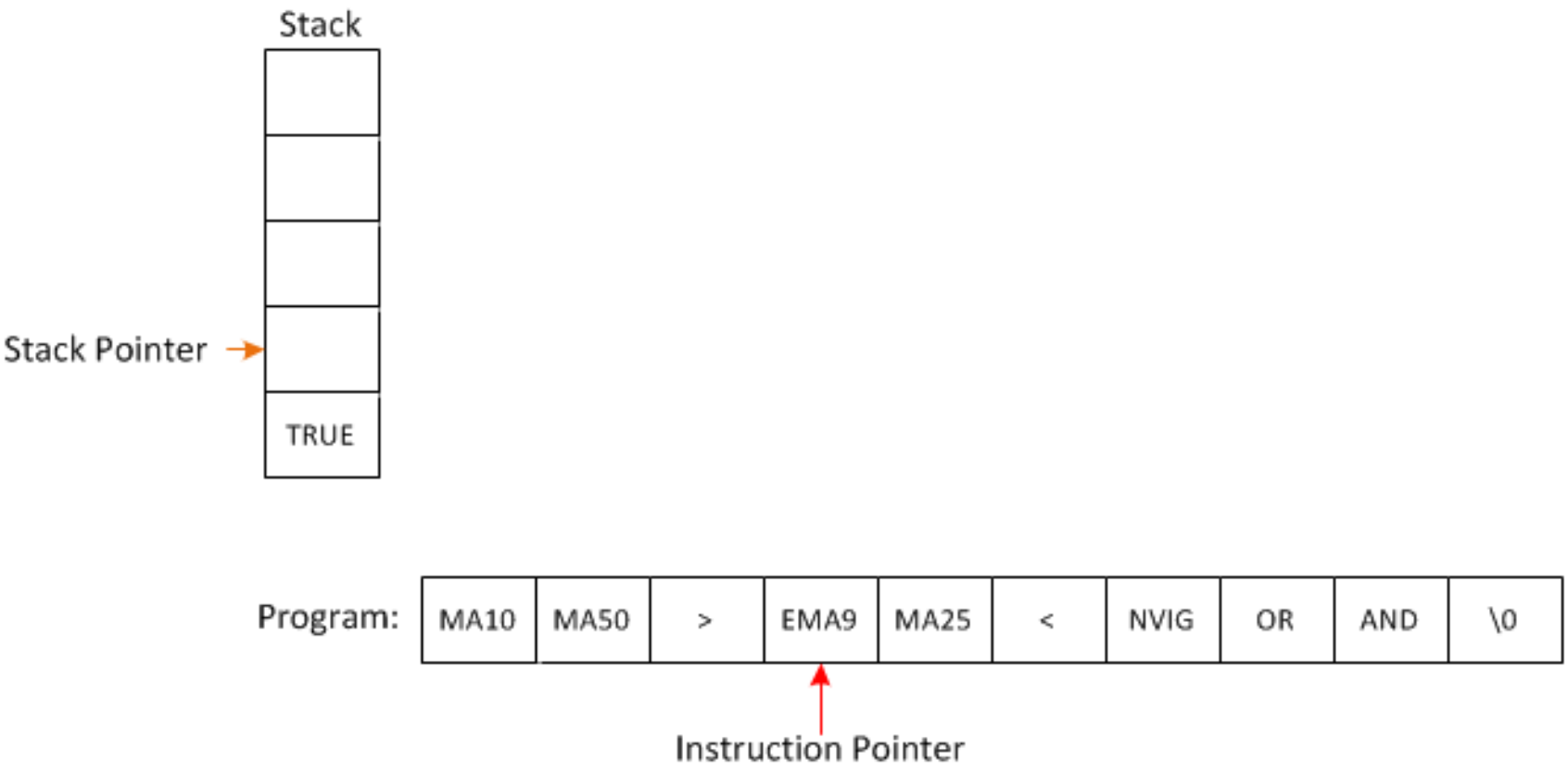
Stack-based Interpreter



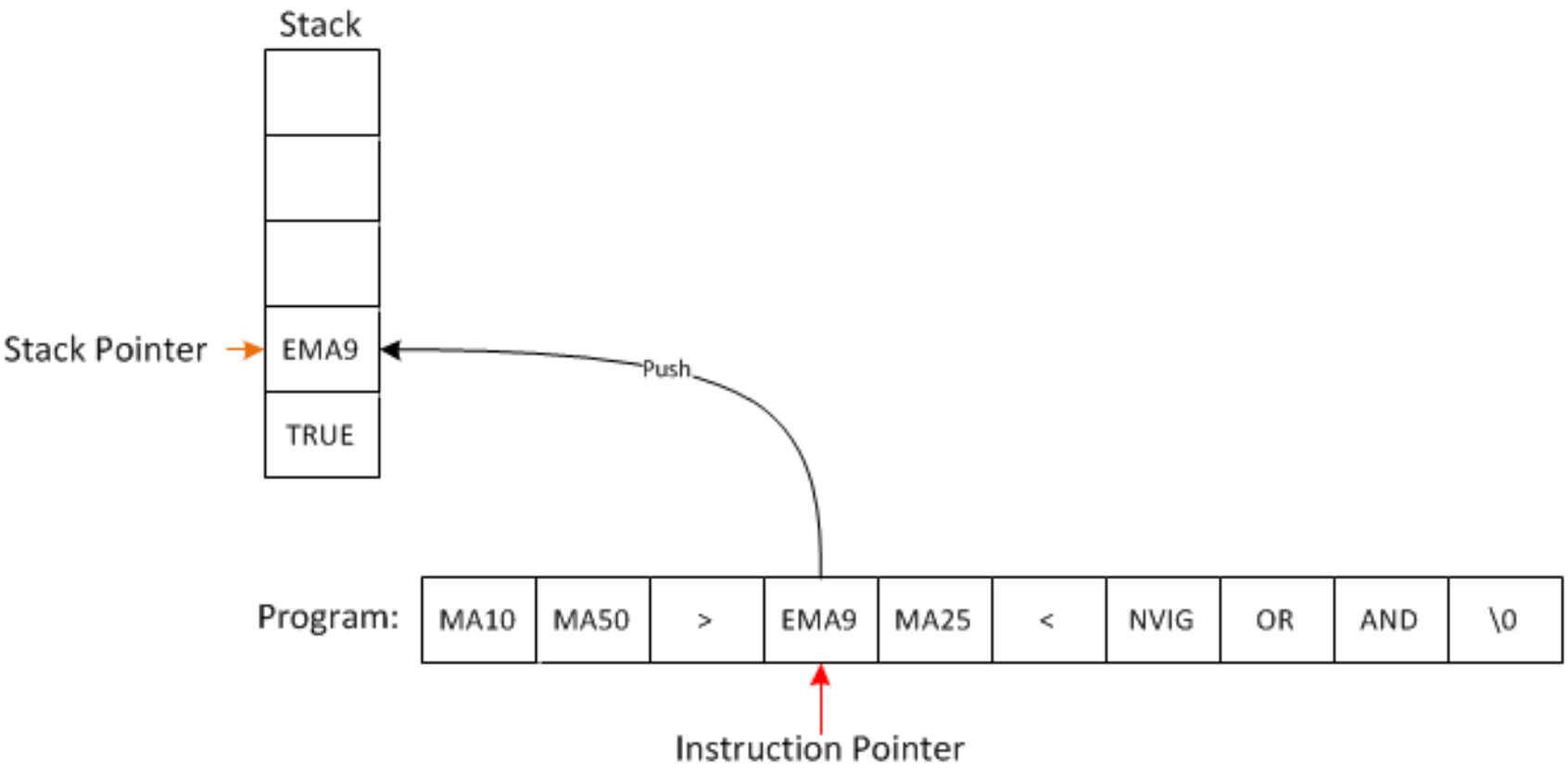
Stack-based Interpreter



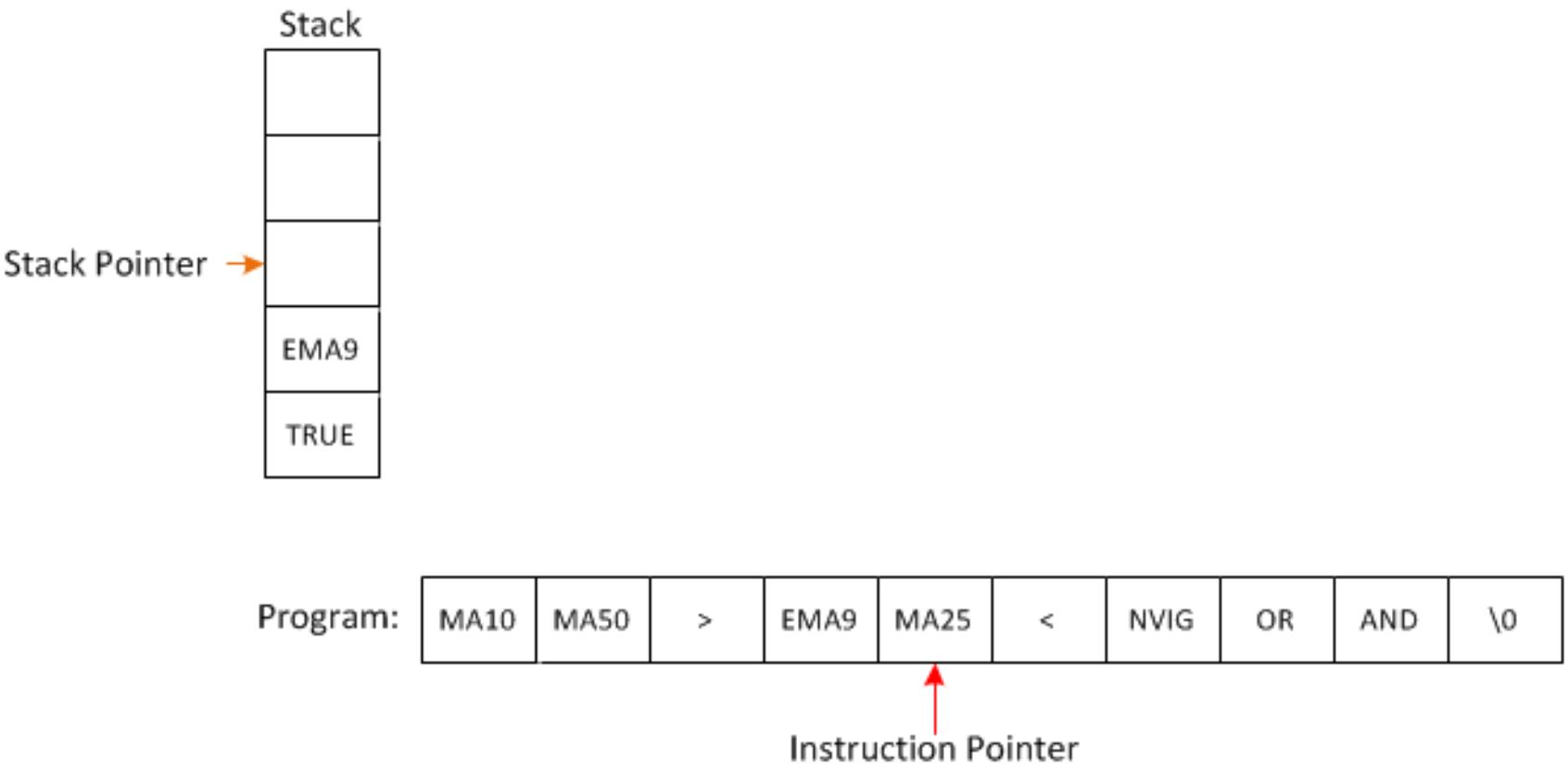
Stack-based Interpreter



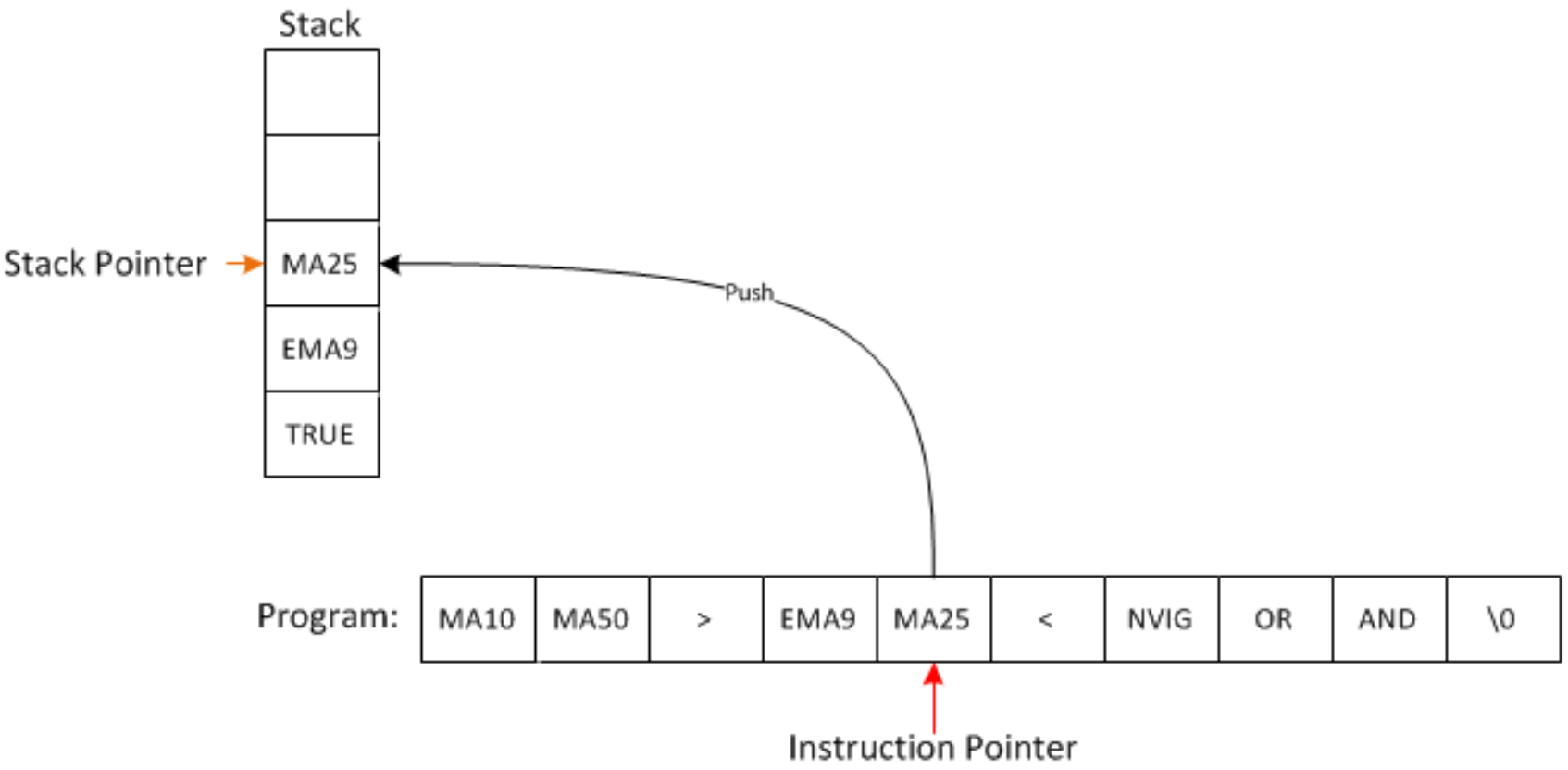
Stack-based Interpreter



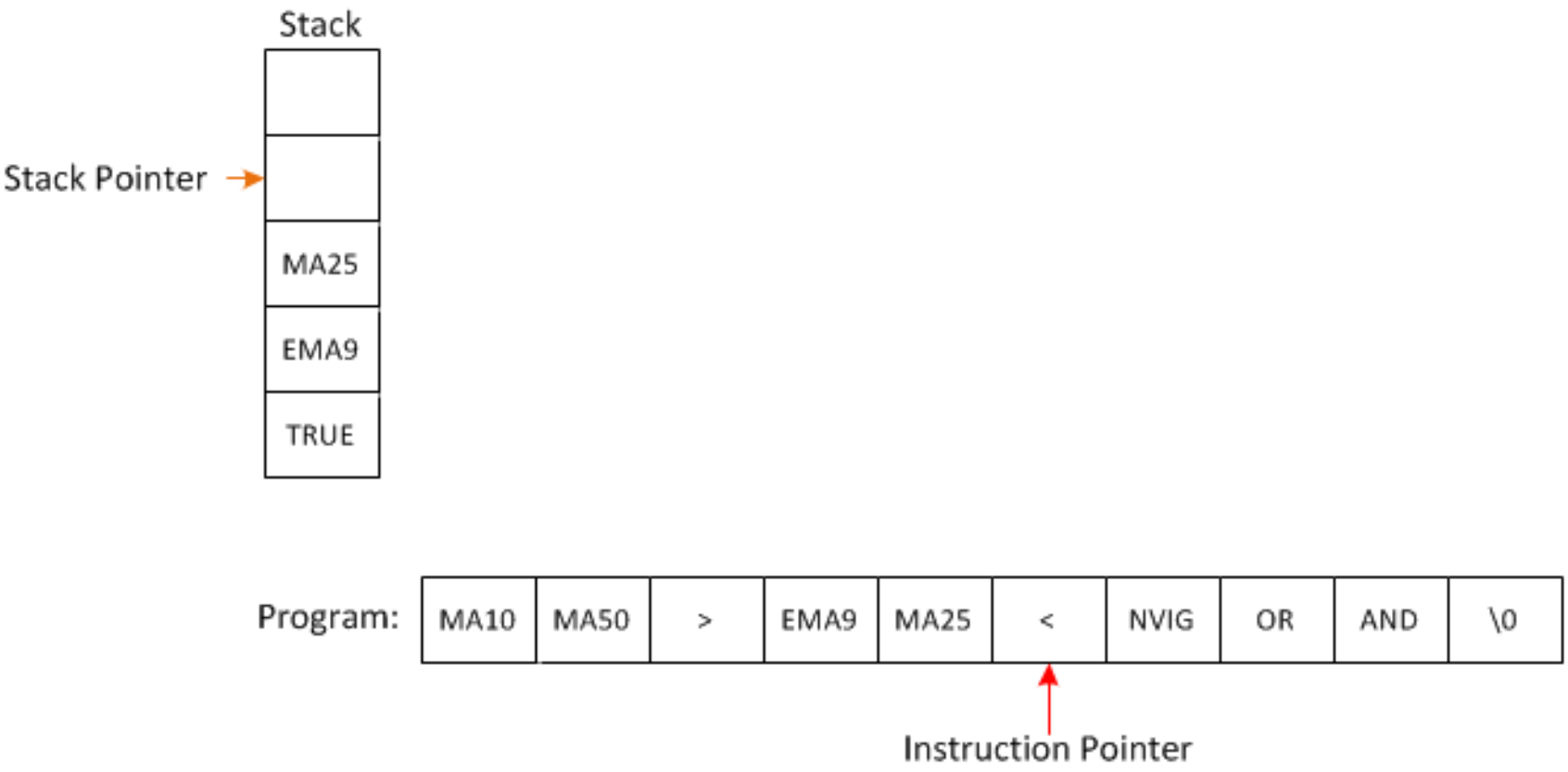
Stack-based Interpreter



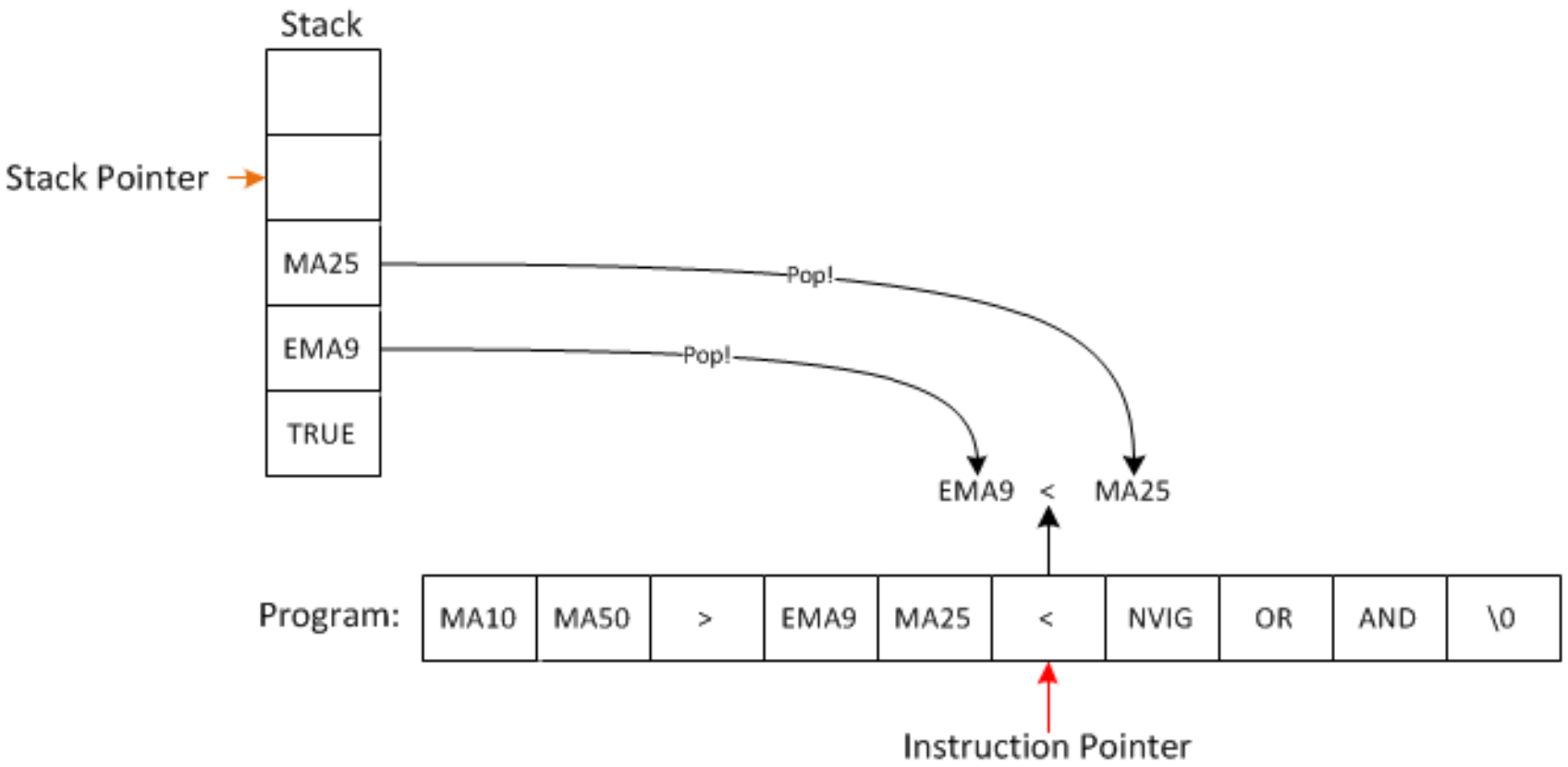
Stack-based Interpreter



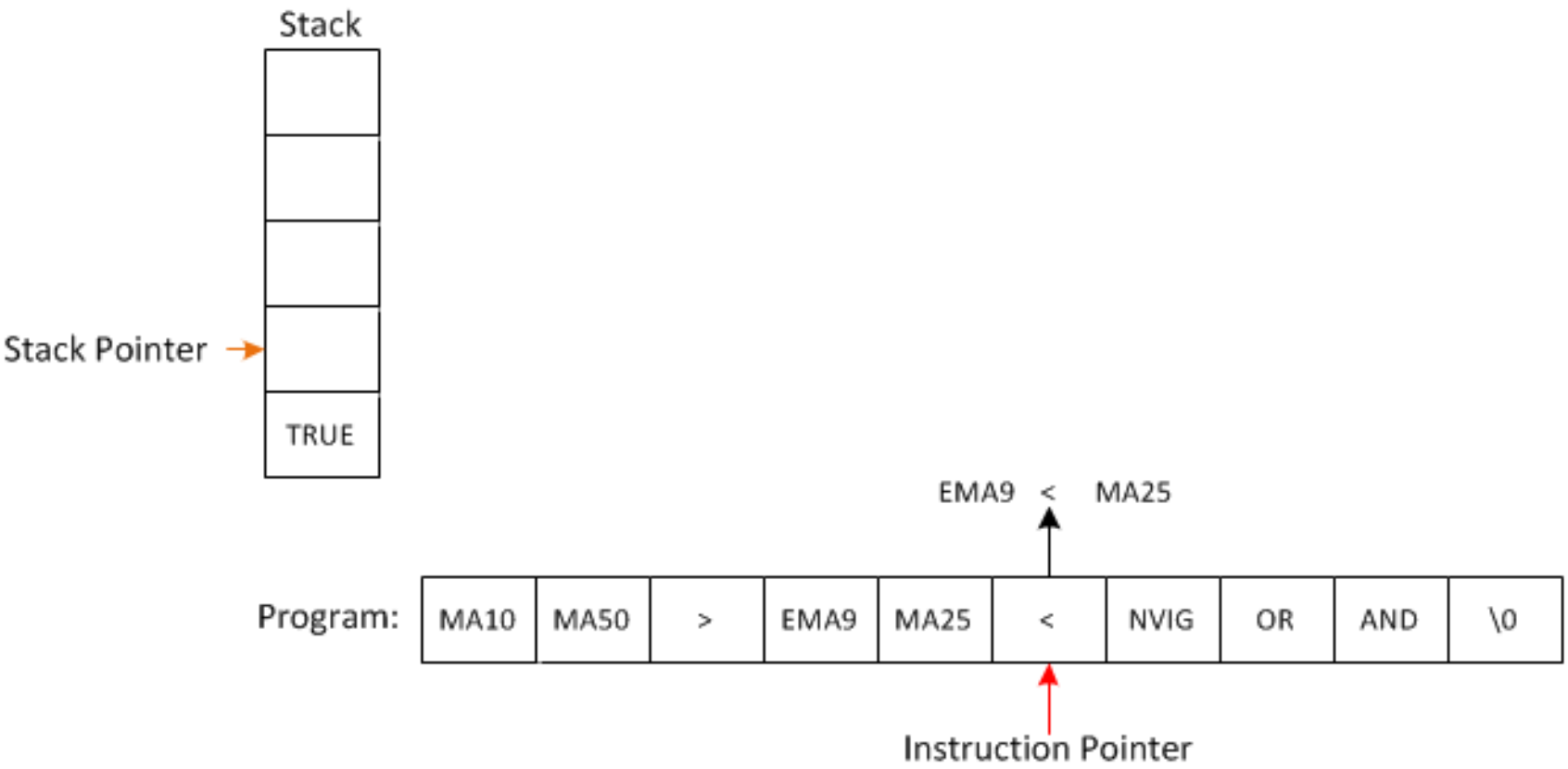
Stack-based Interpreter



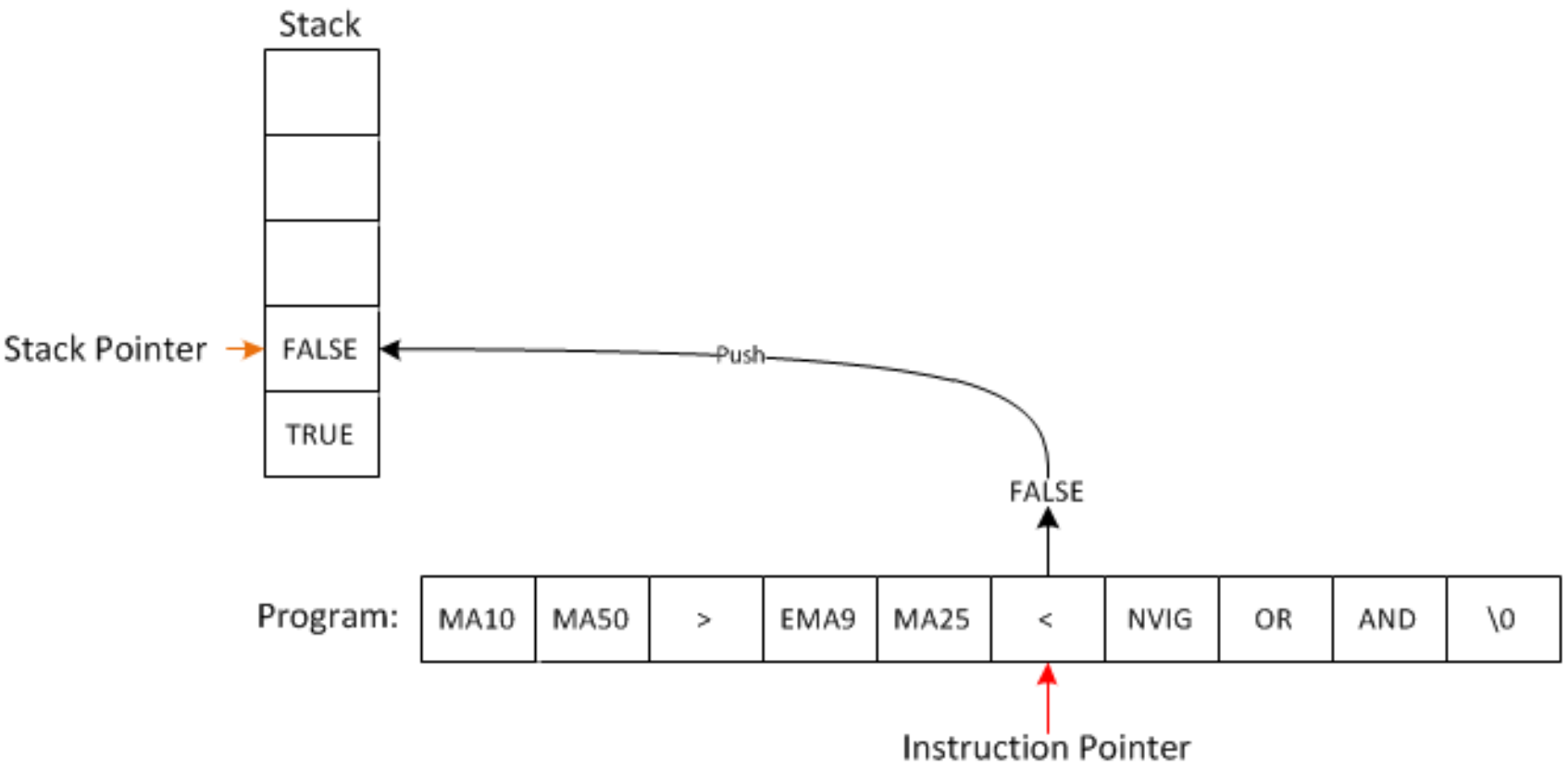
Stack-based Interpreter



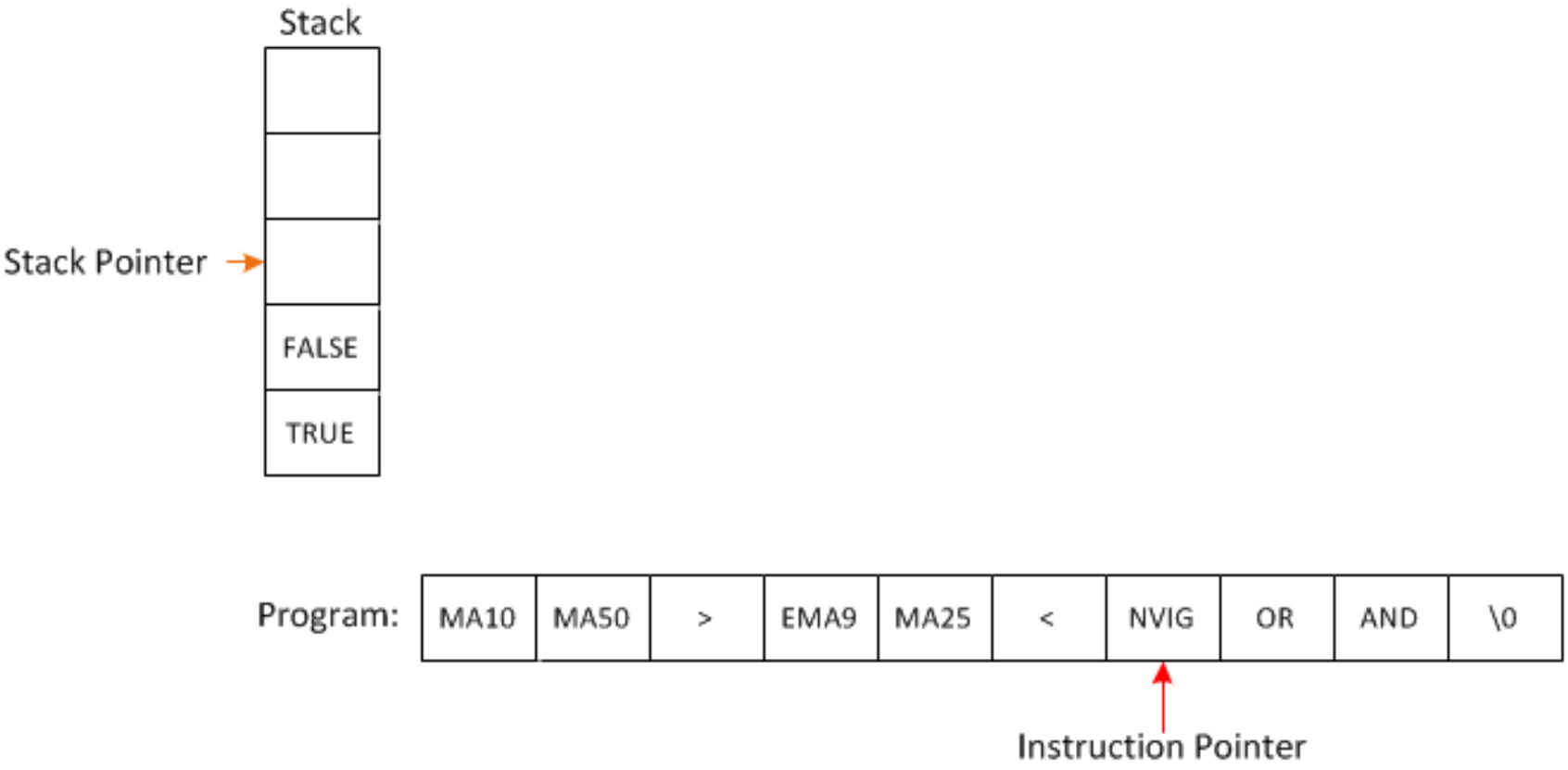
Stack-based Interpreter



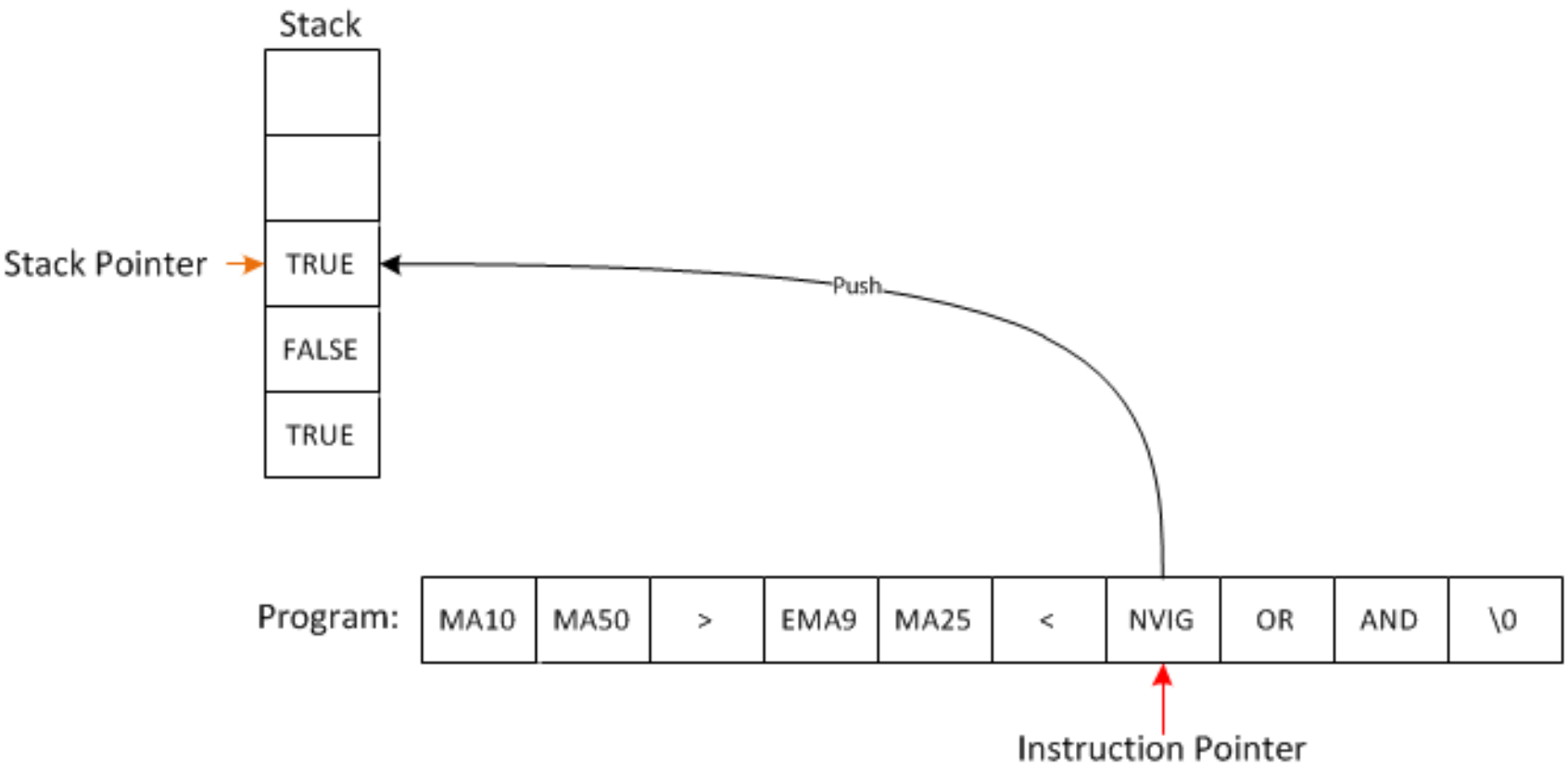
Stack-based Interpreter



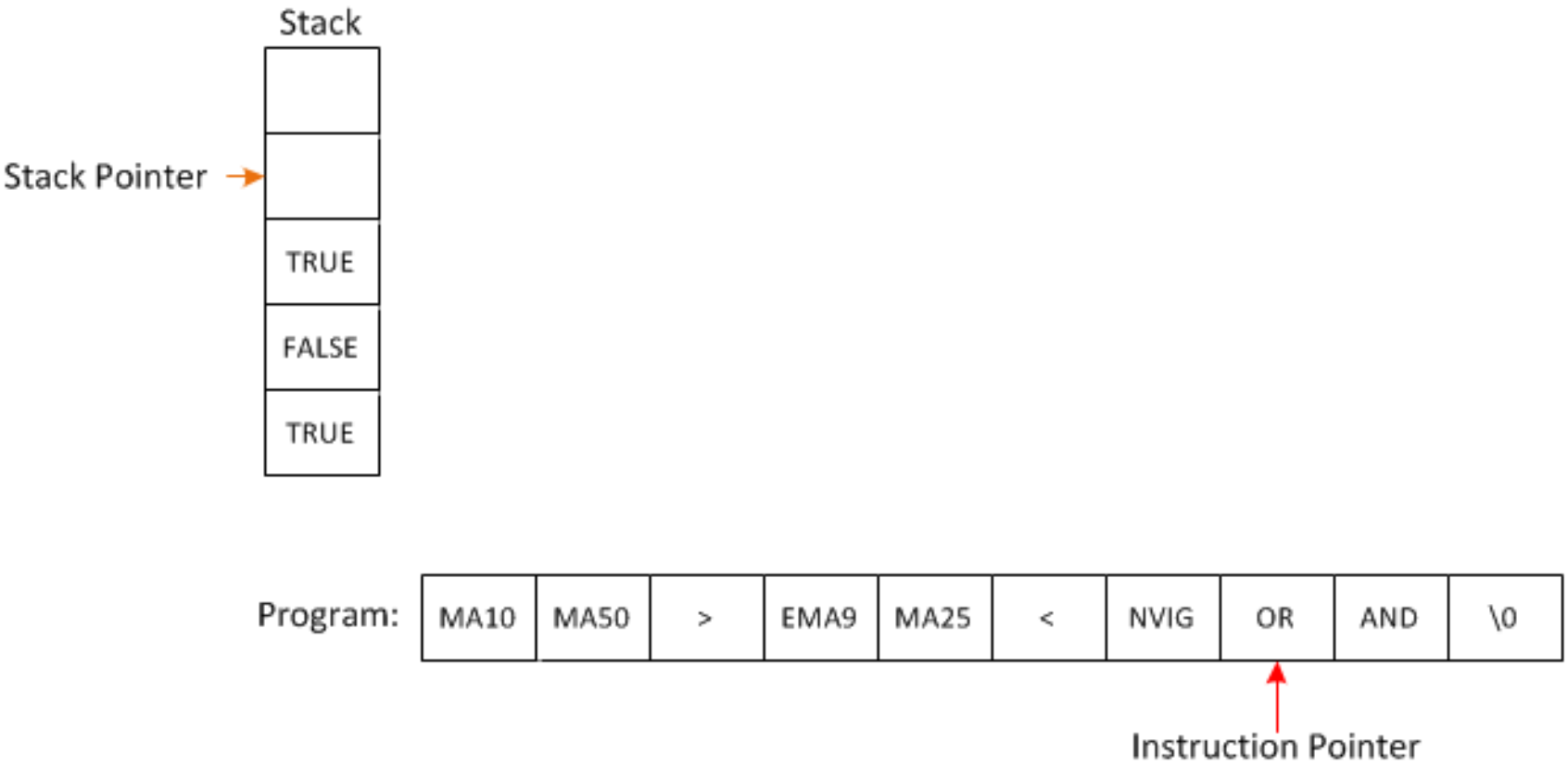
Stack-based Interpreter



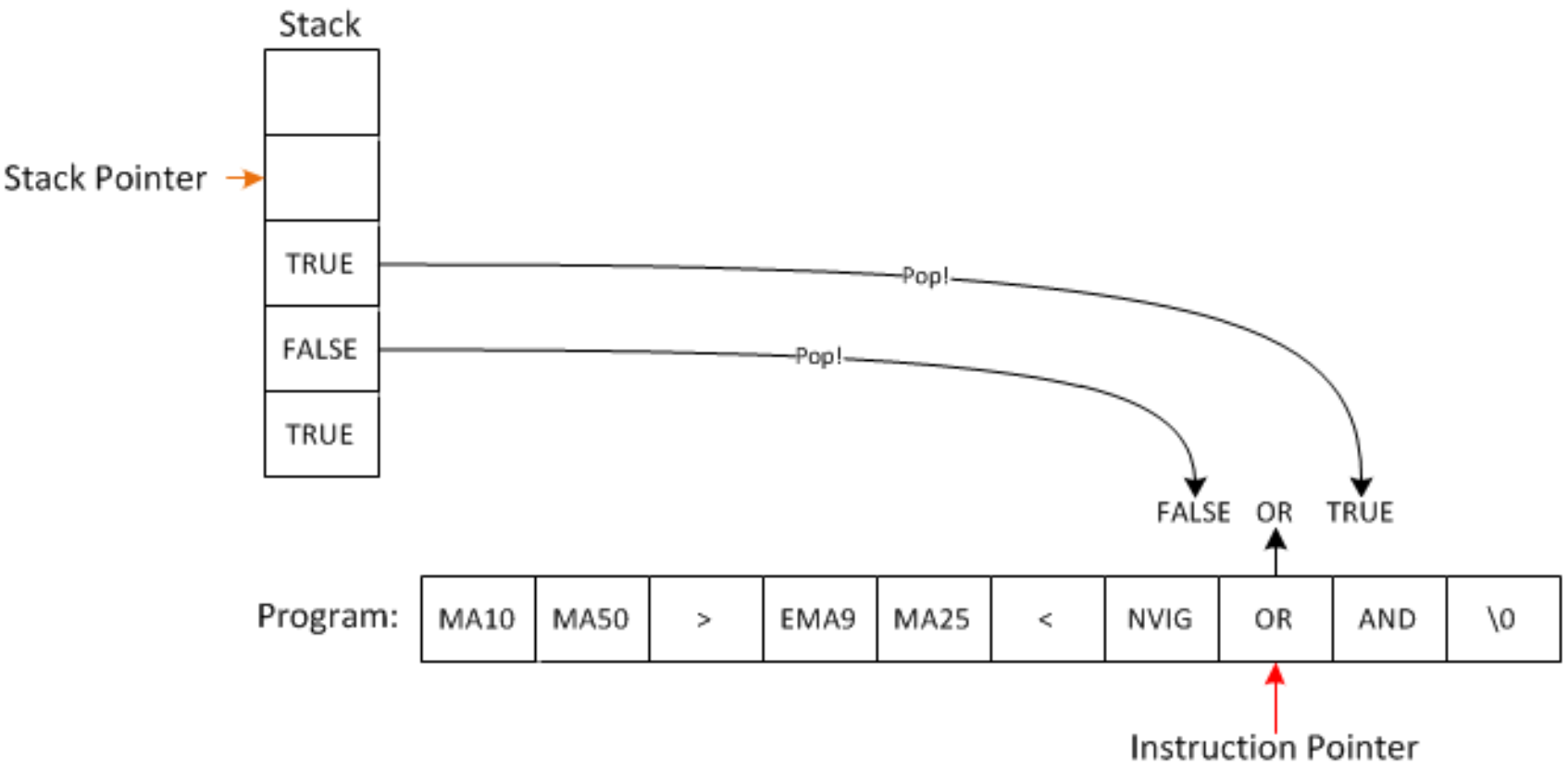
Stack-based Interpreter



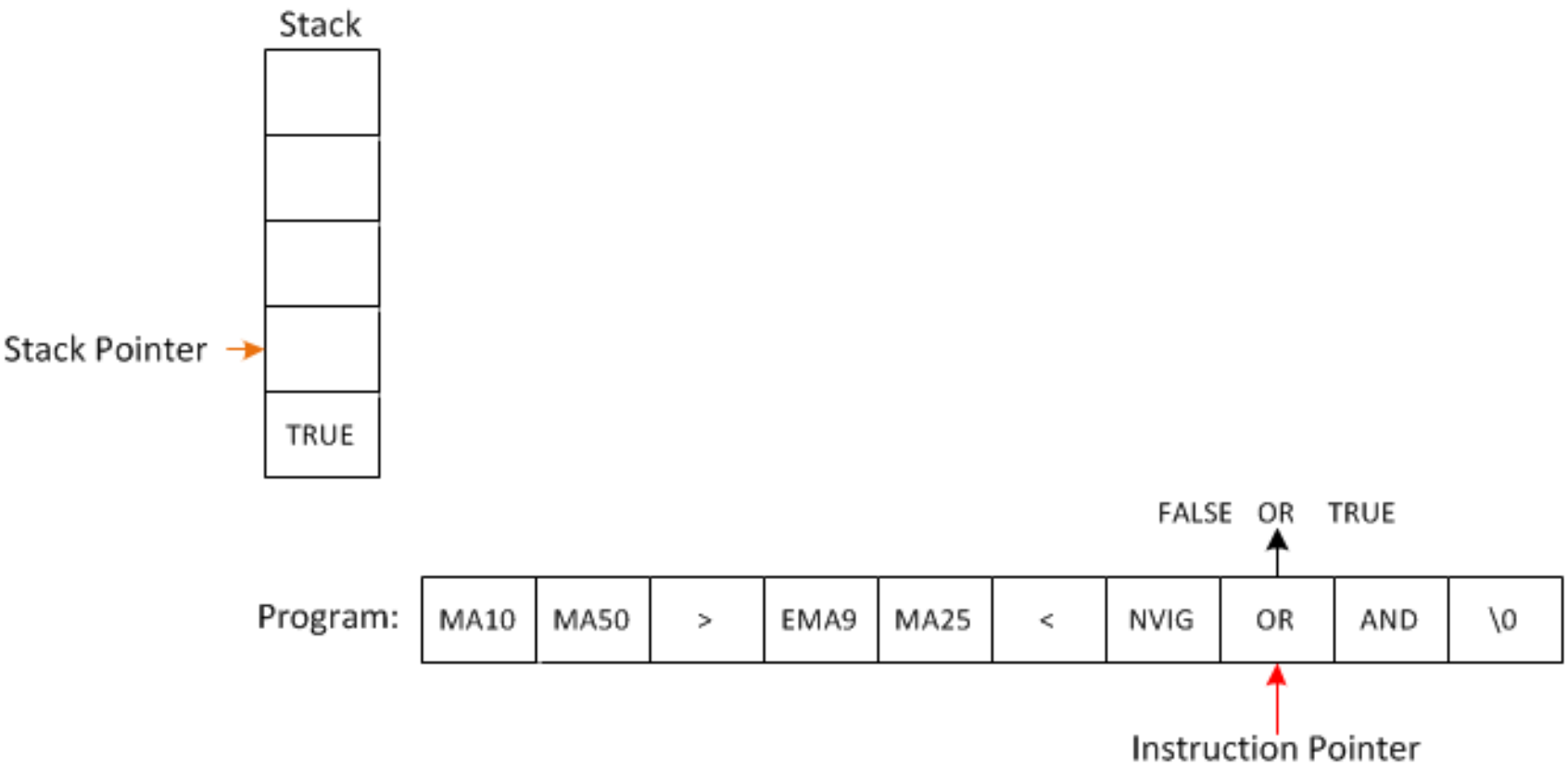
Stack-based Interpreter



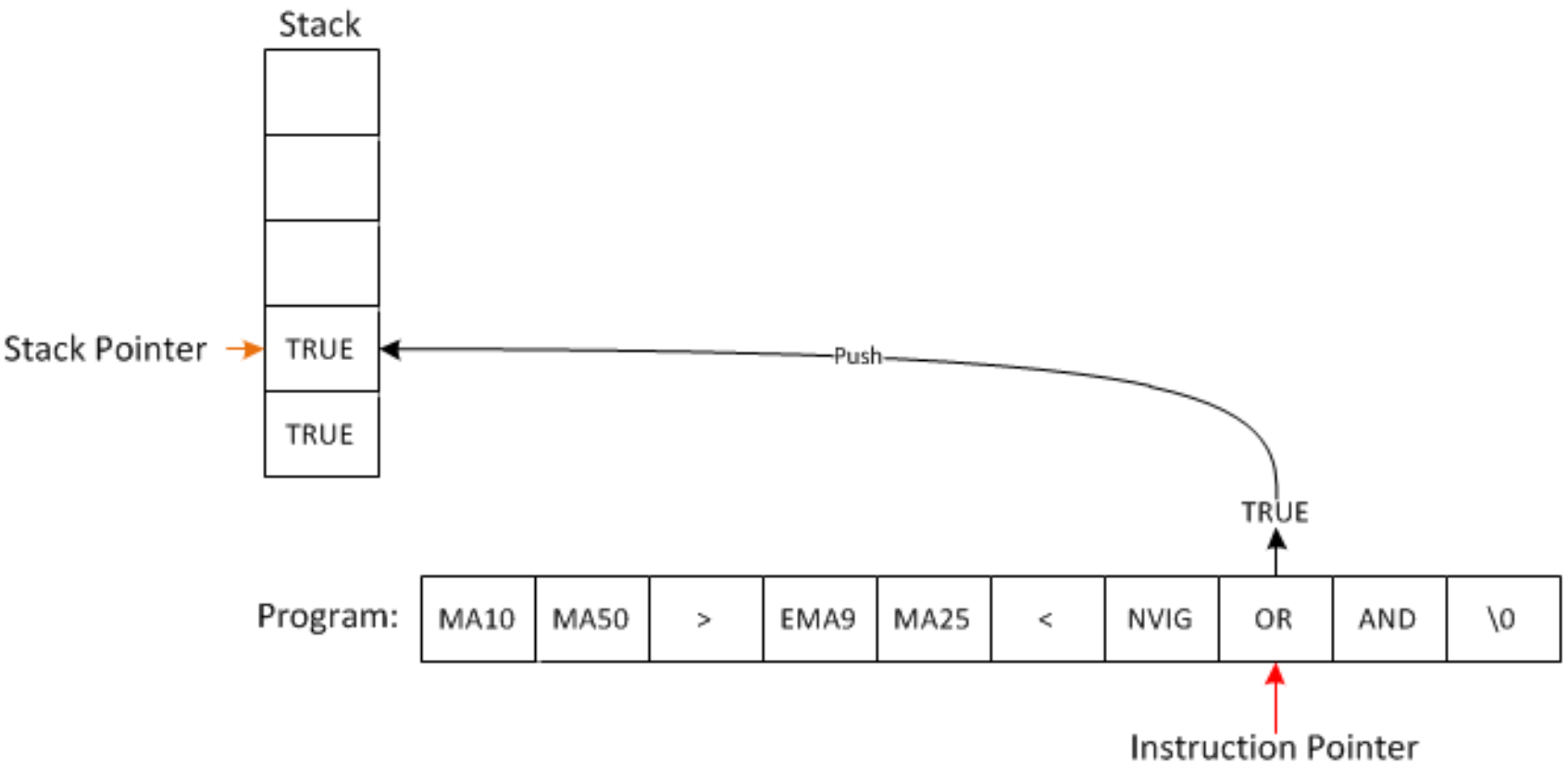
Stack-based Interpreter



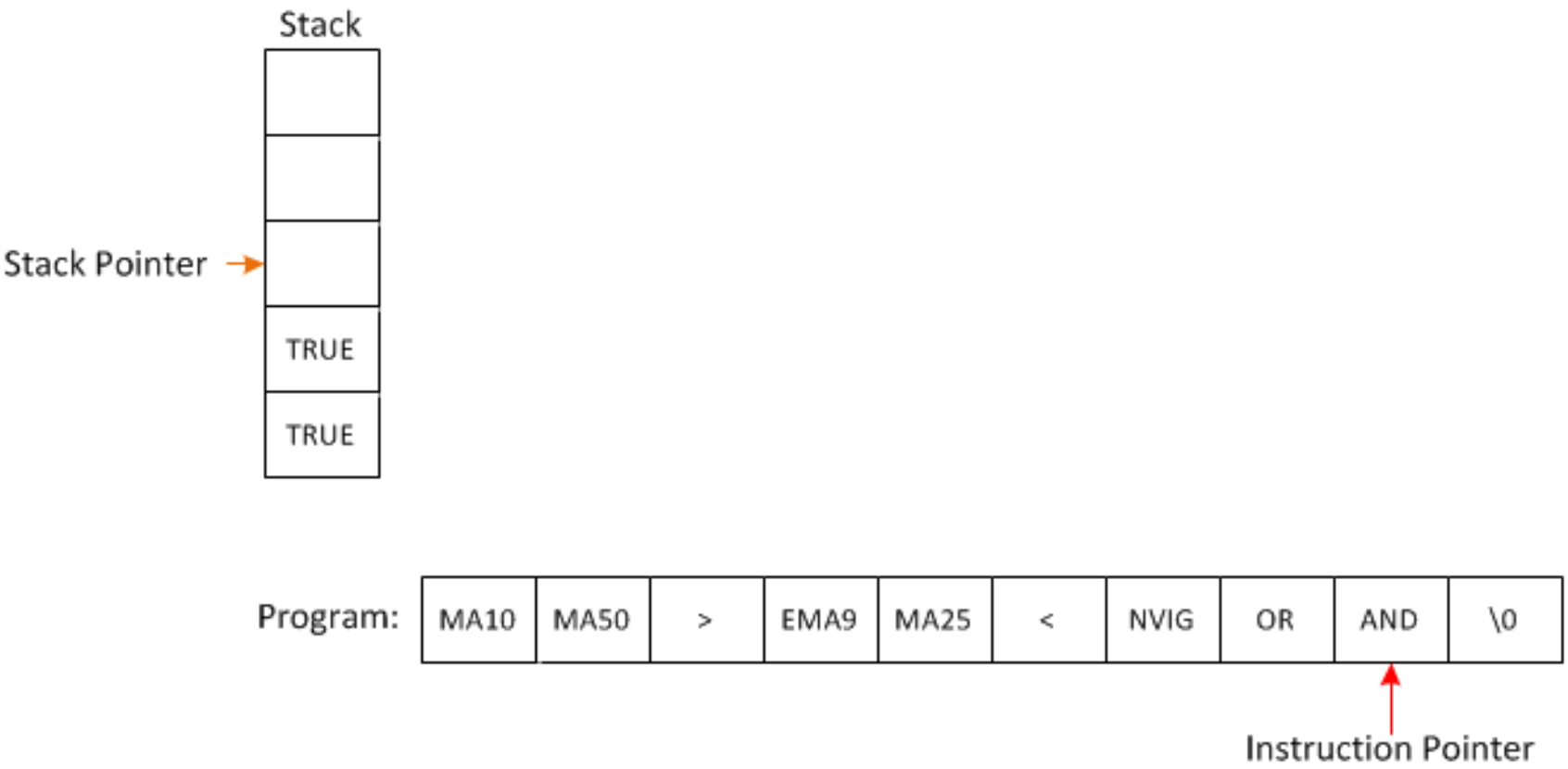
Stack-based Interpreter



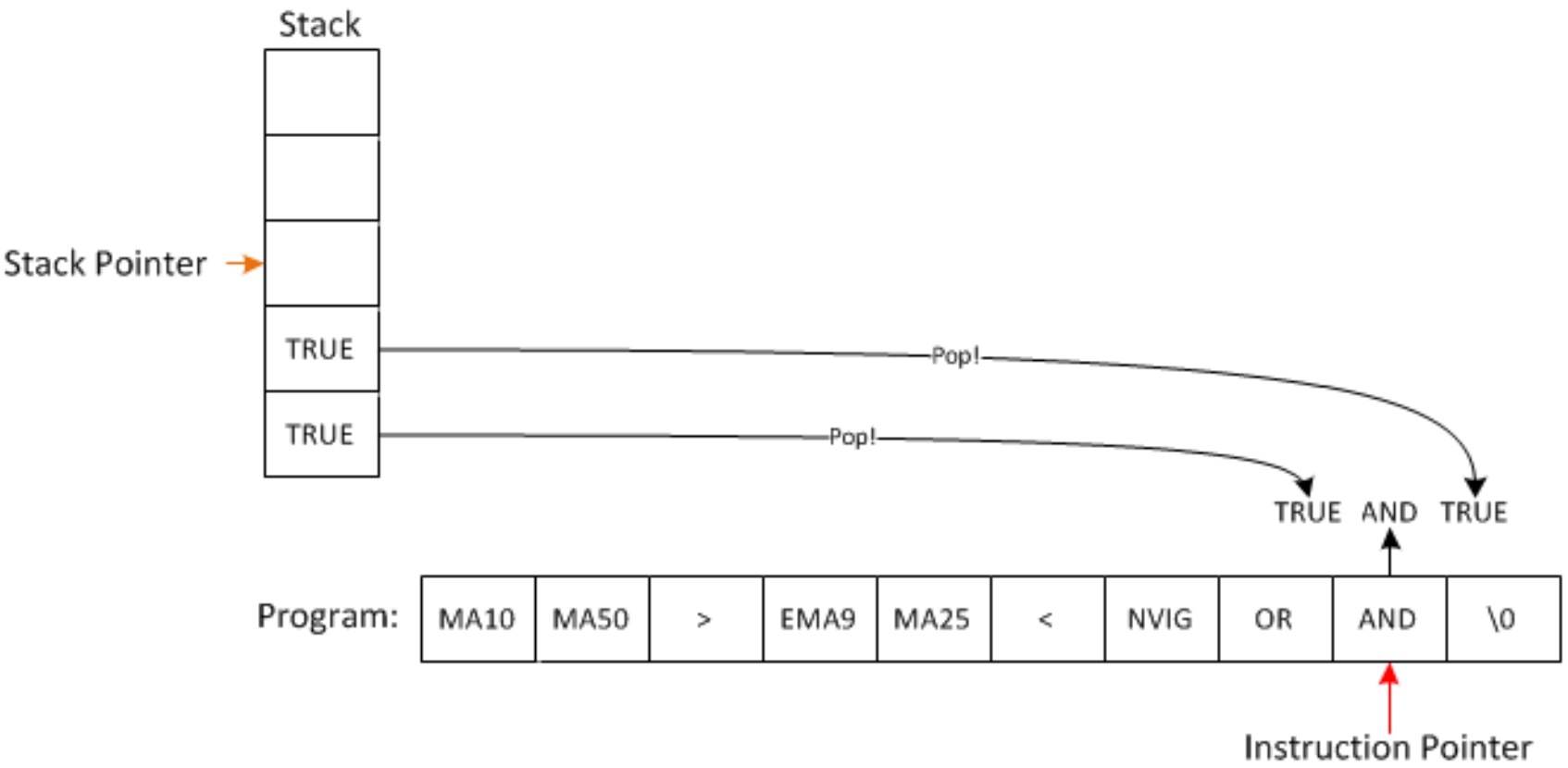
Stack-based Interpreter



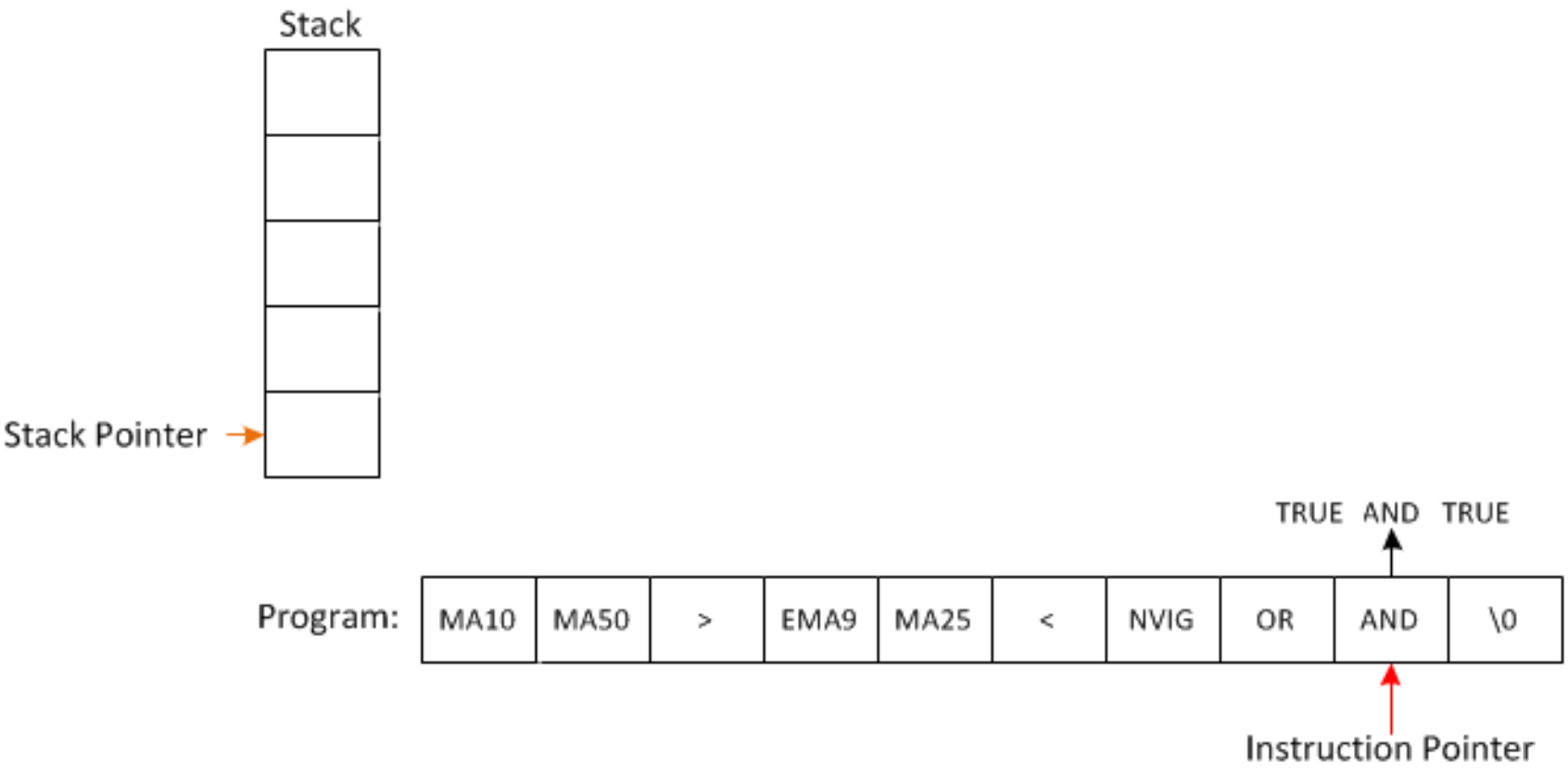
Stack-based Interpreter



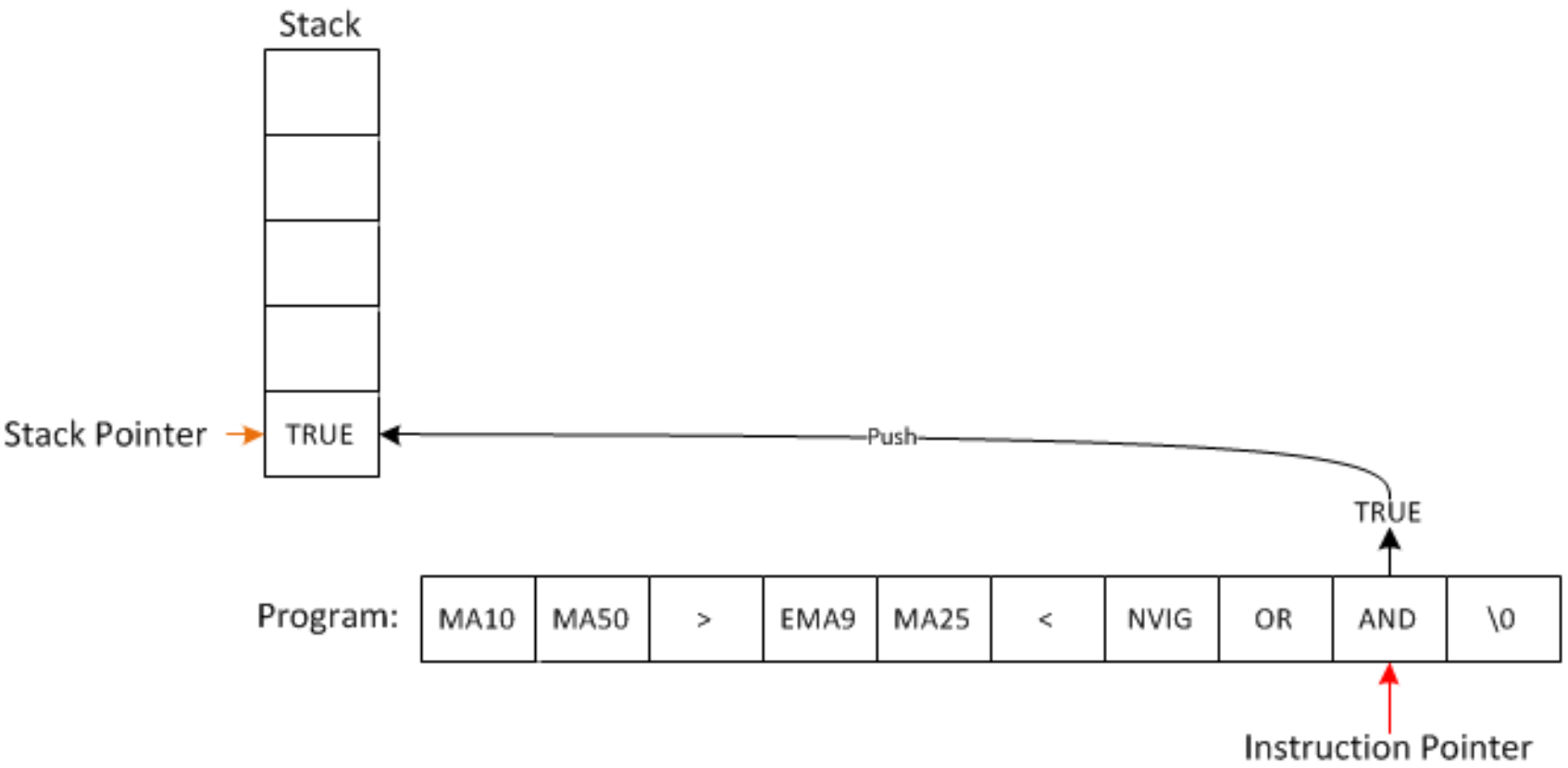
Stack-based Interpreter



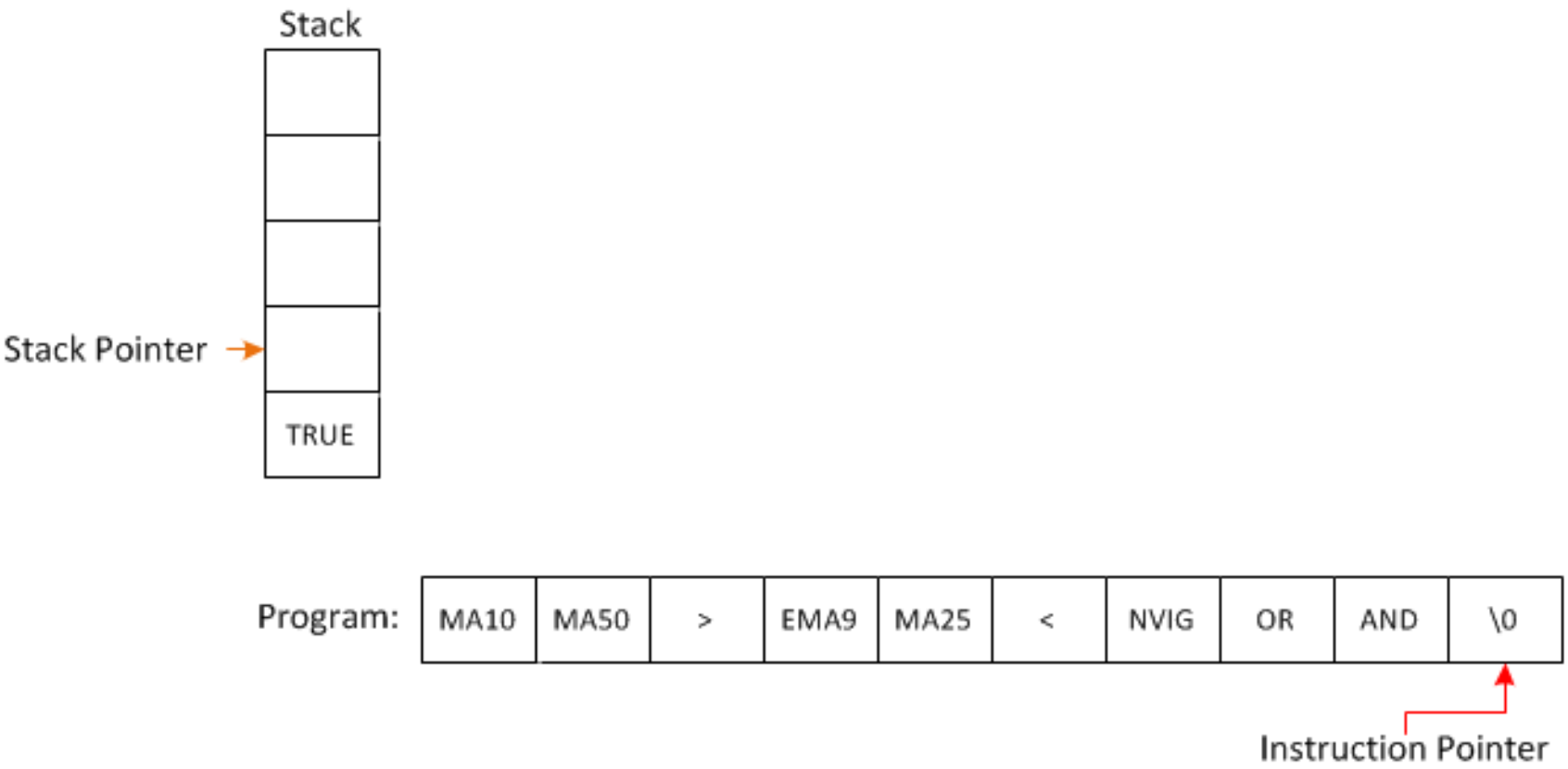
Stack-based Interpreter



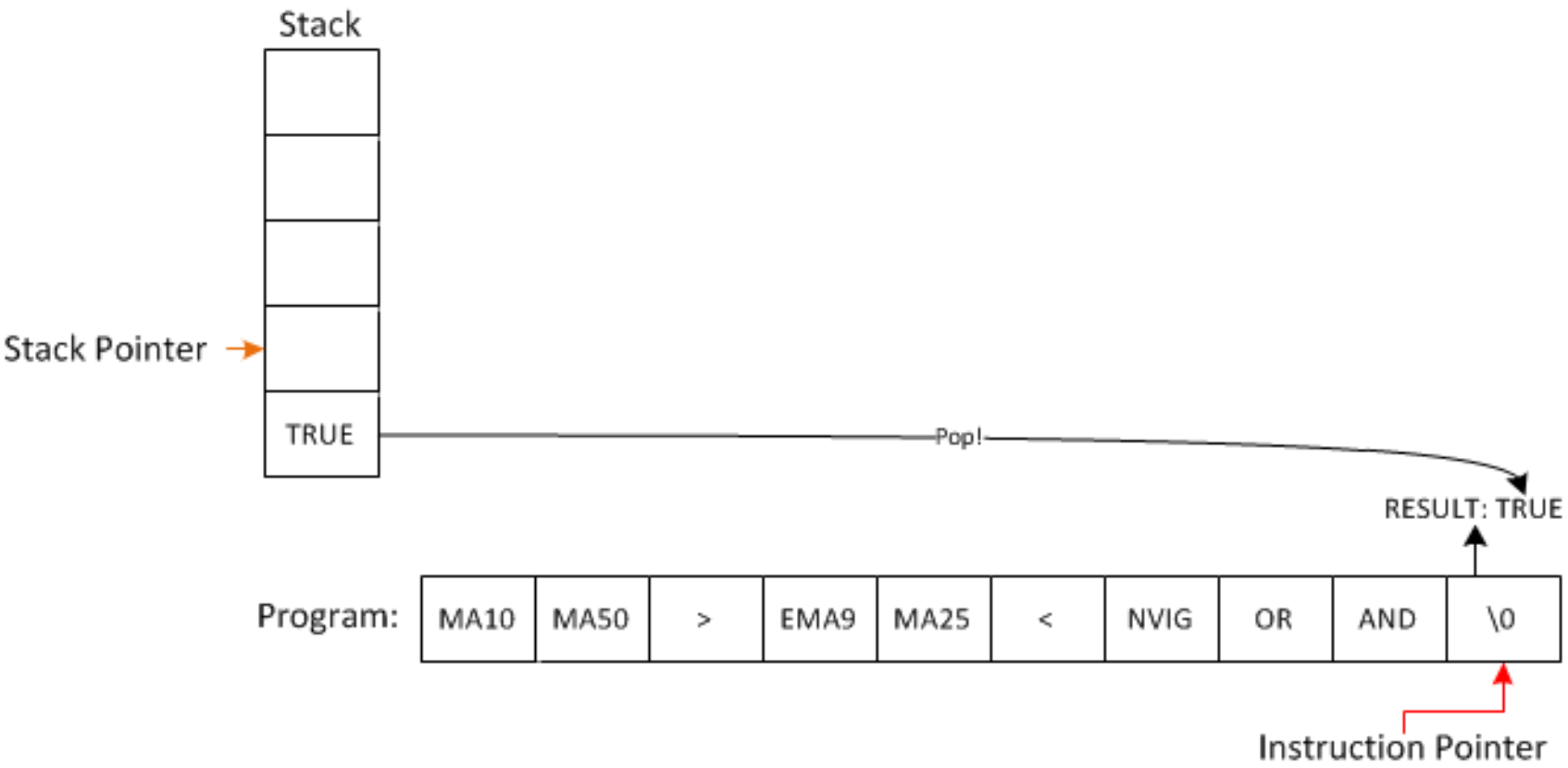
Stack-based Interpreter



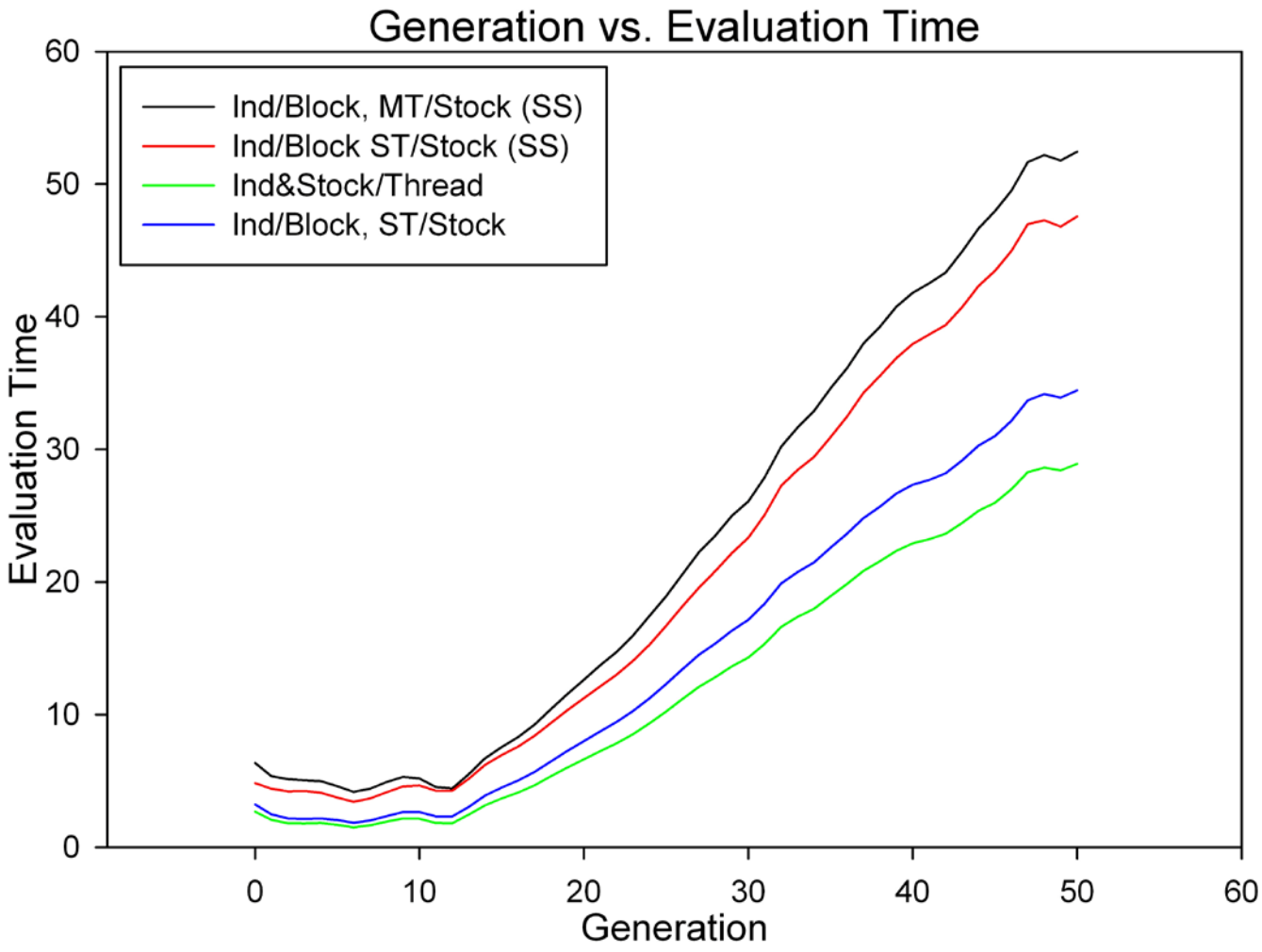
Stack-based Interpreter



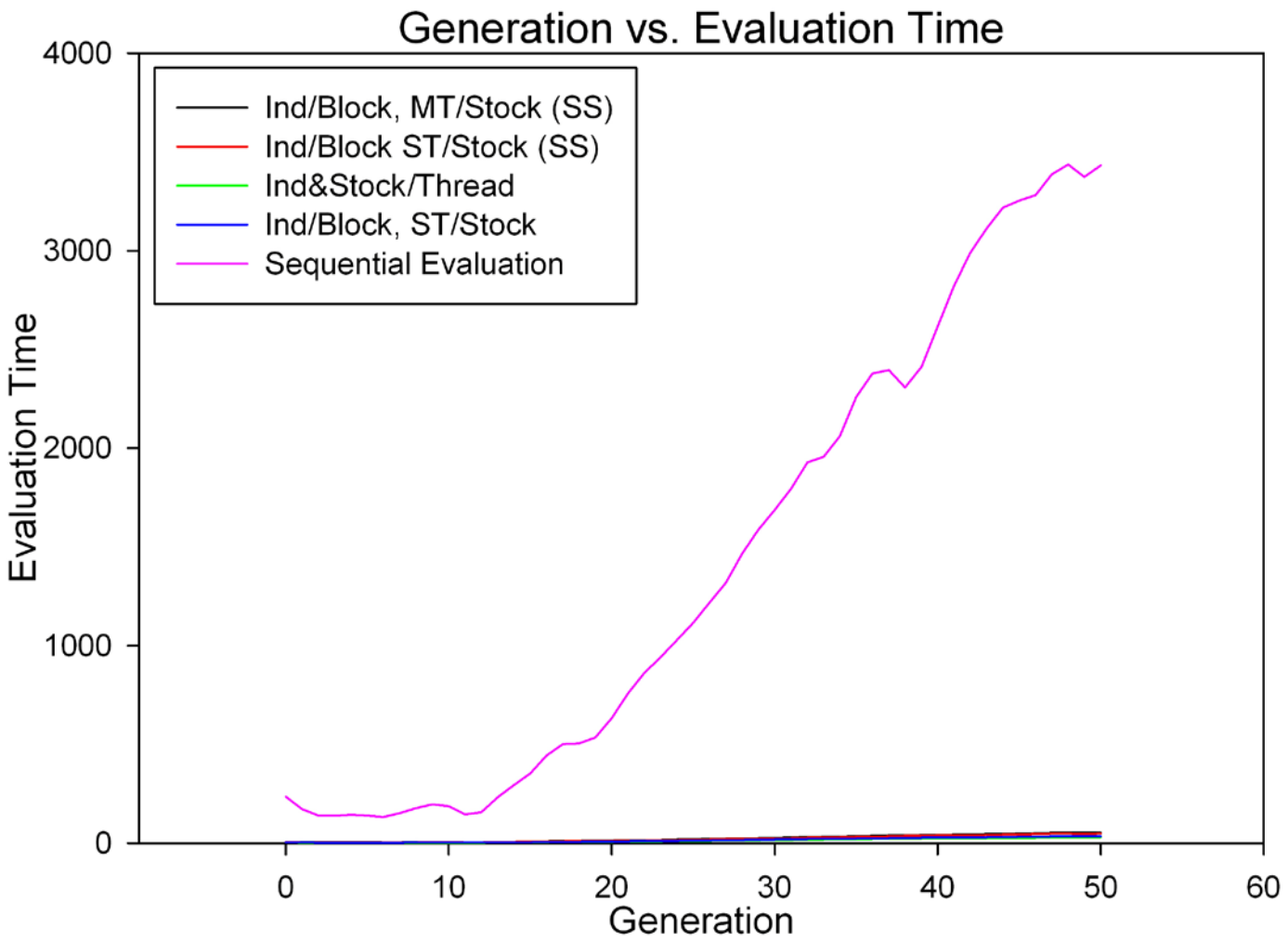
Stack-based Interpreter



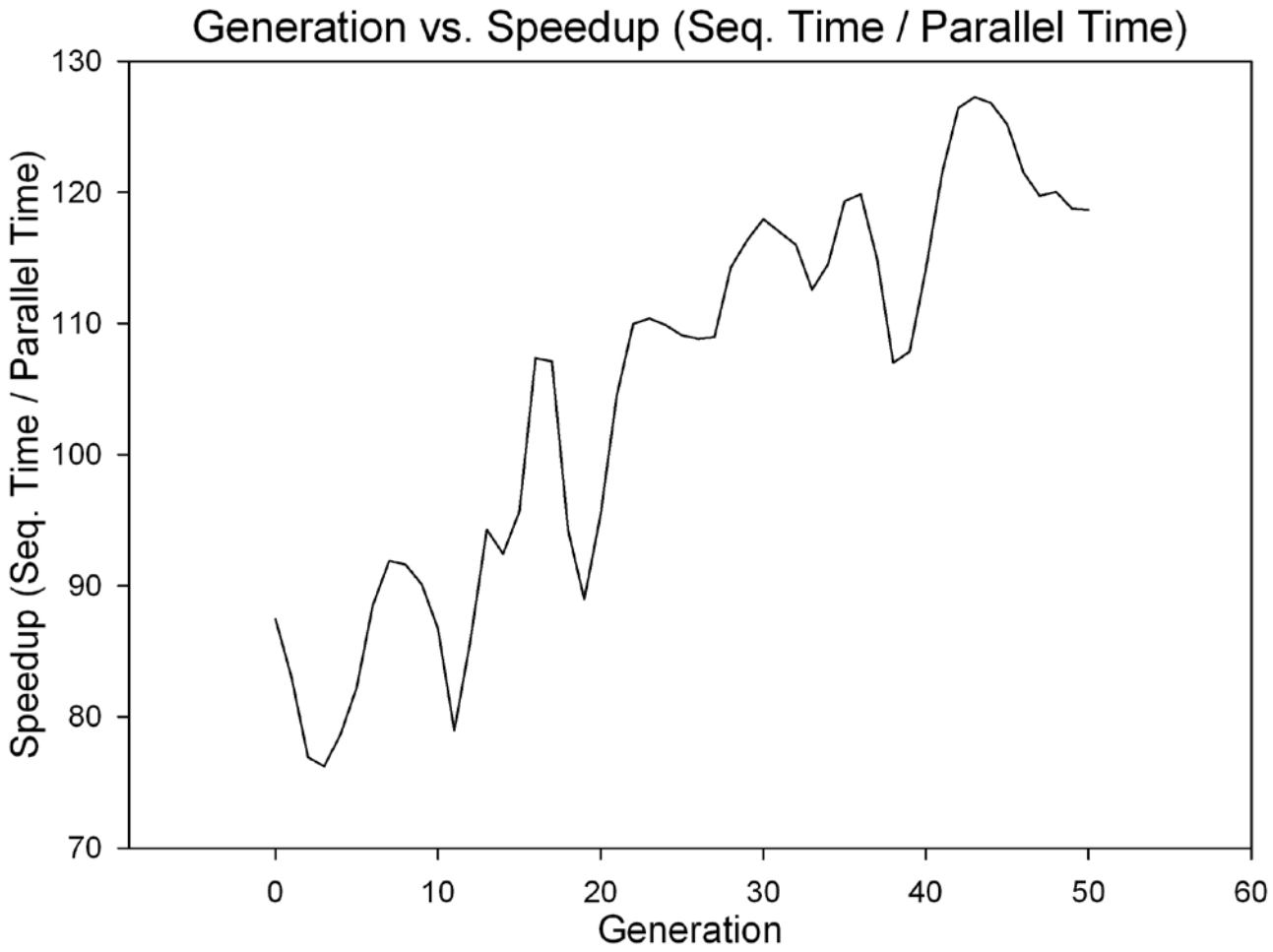
Initial Evaluation Times



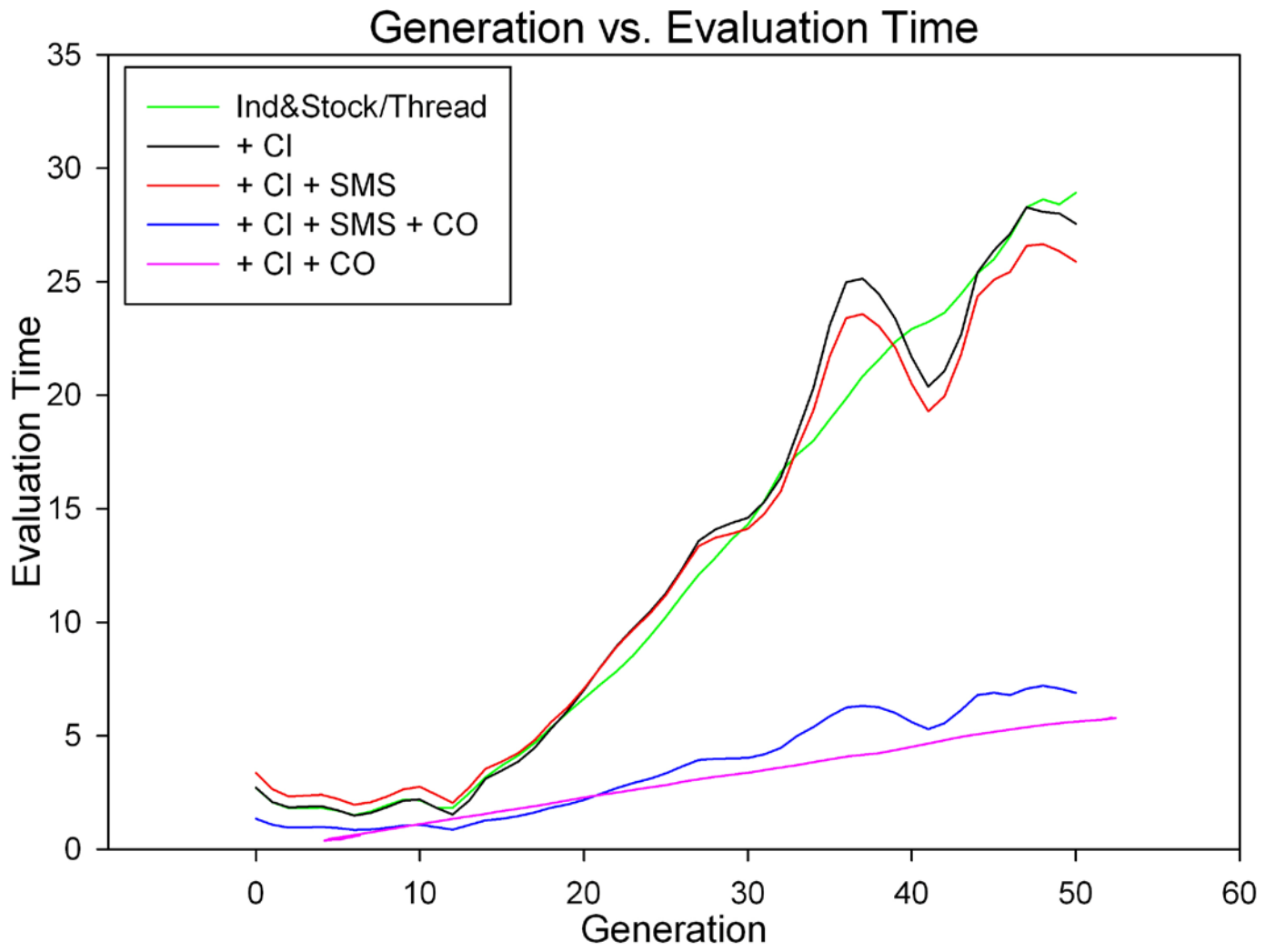
Initial Evaluation Times



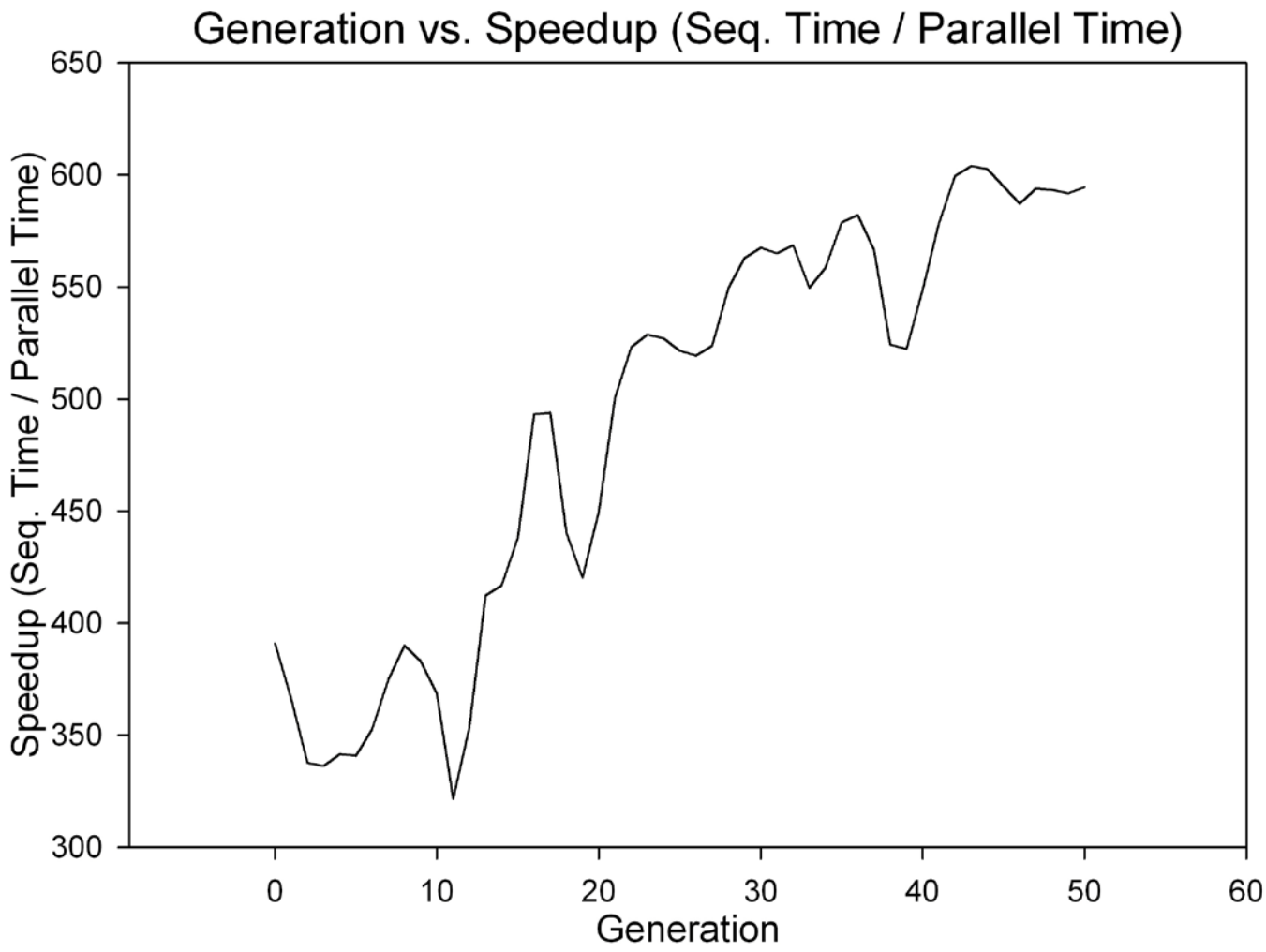
Speedup Comparison



Evaluation Times

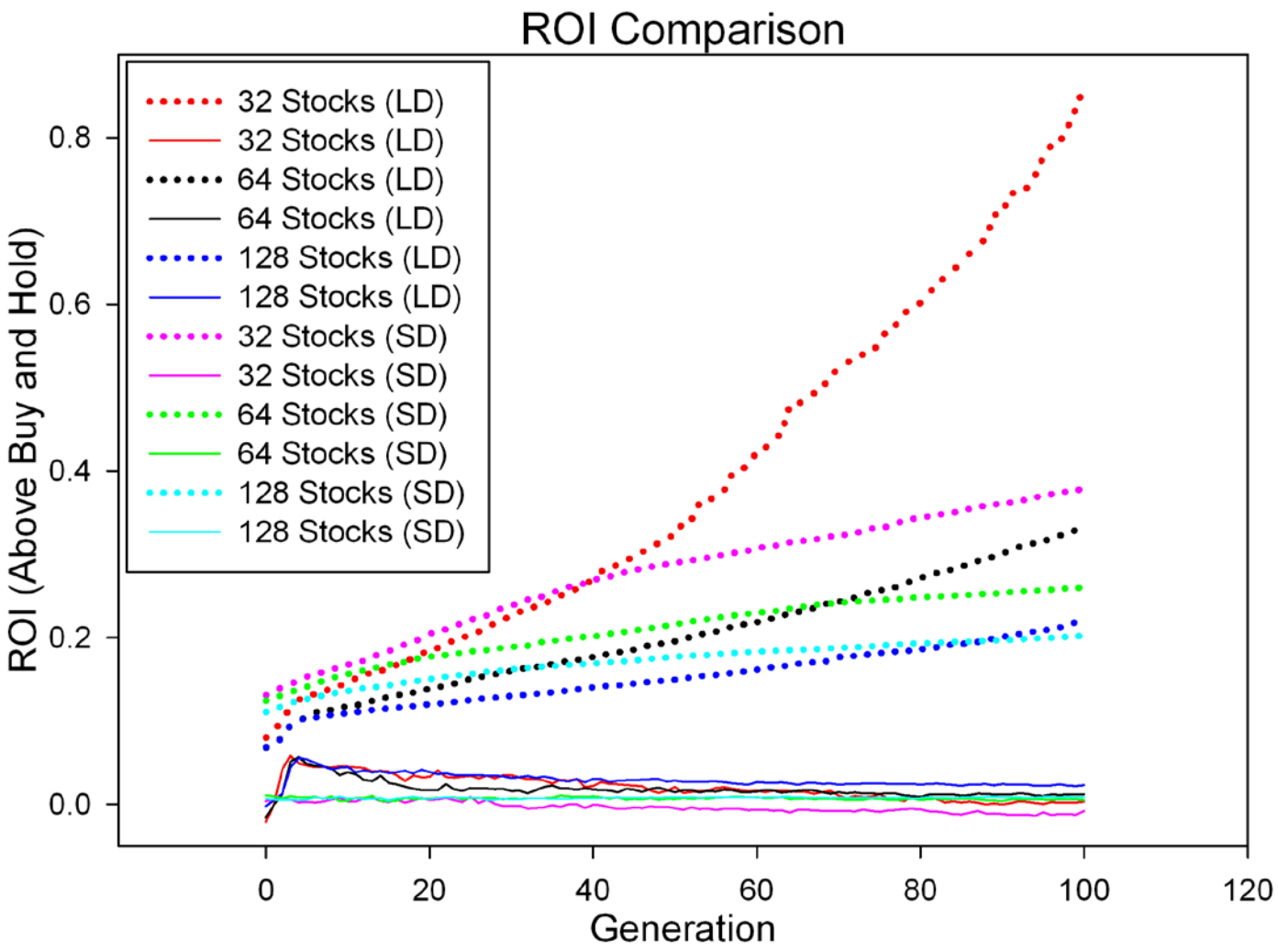


Speedup Results



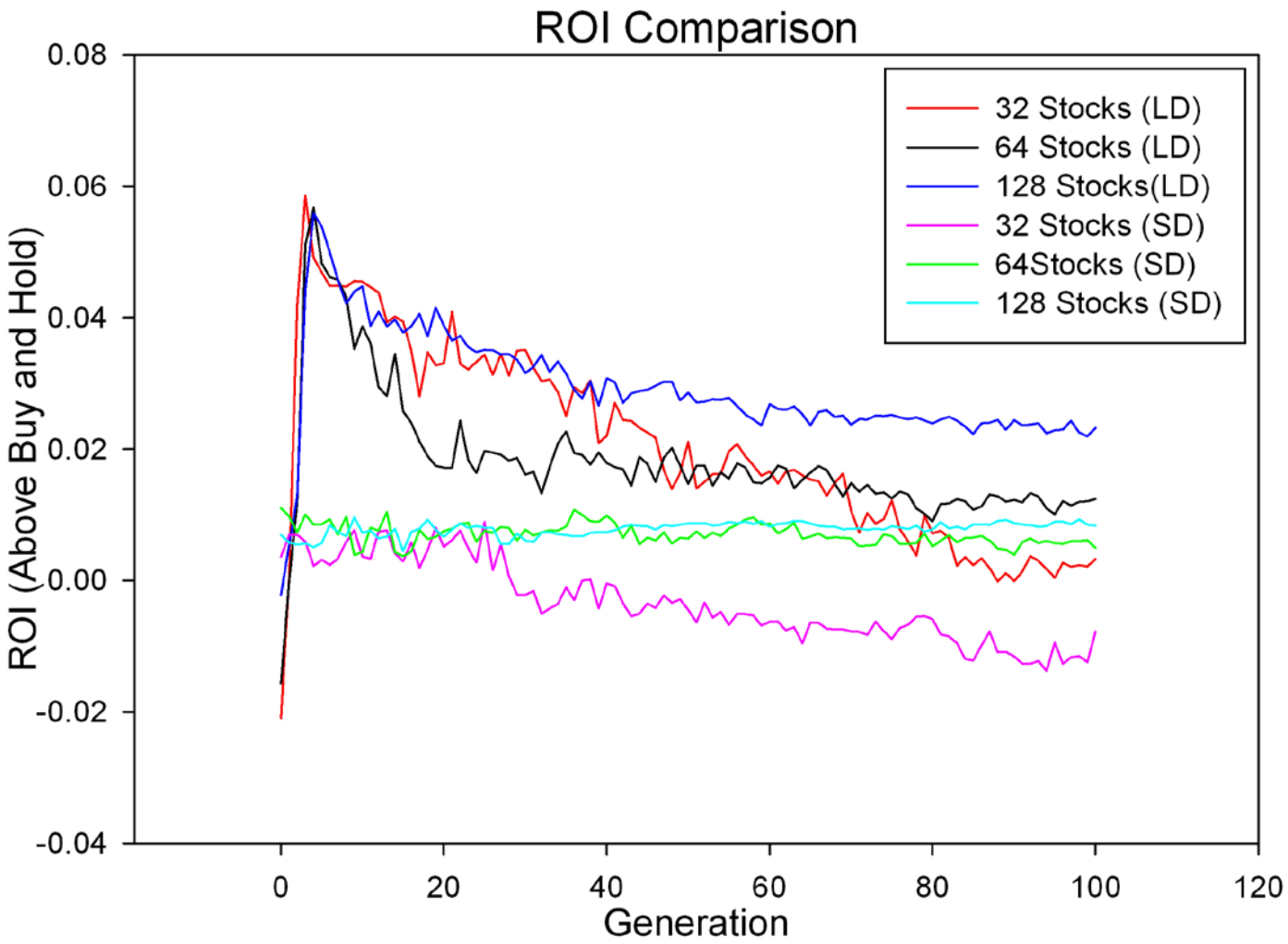
ROI Comparisons

Carleton UNIVERSITY



ROI Comparisons

Carleton UNIVERSITY



Conclusions

Carleton UNIVERSITY

- Evaluation on GPU can greatly increase evaluation speed
- Evaluating more training cases may result in better test case results
- Still a lot of room for improvement

Future Work

Carleton UNIVERSITY

- Stock Trading Specific
  - More advanced indicators
  - Improving GP parameters
  - Improve the stock trading model
- GP on GPU Specific
  - Move entire GP to the GPU
  - Look for more memory optimizations
  - Find other methods of speedup

Thank you!

Author Email:
dmckenne@connect.carleton.ca

Question Sheet

Carleton UNIVERSITY

- What does the biggest problem with using GP to generate stock trading strategies seem to be?
- Why is proper/effective memory layout important when performing GP on GPU devices?
- Name one way to decrease premature convergence when using GP.