Usage Patterns of Trading Rules in Stock Market Trading Strategies Optimized with Evolutionary Methods

Krzysztof Michalak[1], Patryk Filipiak[2], Piotr Lipinski[2]

[1]Institute of Business Informatics,
Wroclaw University of Economics, Wroclaw, Poland
krzysztof.michalak@ue.wroc.pl

[2]Institute of Computer Science,
University of Wroclaw, Wroclaw, Poland
{patryk.filipiak,lipinski}@ii.uni.wroc.pl

Presentation Plan

- Trading rules
- Trading experts
- Evolutionary optimization
- Analysis of Trading Rule Sets
- Experiments
- Results
- Conclusions

Trading Rules

- Based on
  - stock price
  - volume
  - technical indicators

- Parameters
  - period
  - threshold
  - moving average type (SMA / EMA)

Trading Rules – Example

Trading Rules

- Technical Indicators
  - Chaikin Oscillator
  - Ease of Movement Value (EMV)
  - K-Stochastic
  - MACD
  - Moving Averages (MA)
  - Overbought / Oversold
  - Rate of Change (RoC)
  - Relative Strength Index (RSI)
  - Top / Bottom Reversal (TBR)
  - Williams Oscillator
  - ...

Trading Experts

Trading expert parameters:

$$weight_1, par_{1,1}, \ldots, par_{1,k_1}, \ldots, \underbrace{weight_n}_{n\text{-th rule weight}}, \overbrace{par_{n,1}, \ldots, par_{n,k_n}}^{n\text{-th rule parameters}}, \Theta_{buy}, \Theta_{sell}$$

where:

$weight_i$ - i-th rule weight,
$par_{i,j}$ - j-th parameter of the i-th rule,
k_i - number of parameters for the i-th rule,
Θ_{buy}, Θ_{sell} - decision thresholds.

Evolutionary Optimization (1/3)

- Intra-day trading simulation
  - Sell everything at the end of the day even if no „sell” signal
- Commission: 0.4%
- Trading decisions
  - specimens contain trading expert parameters
  - individual rule parameters are extracted from the genotype
  - rules generate their „buy” and „sell” signals
  - weighted sum of signals is calculated
  - decision thresholds are applied

Evolutionary Optimization (2/3)

Algorithm 1. Evolutionary algorithm used for optimizing trading experts

N_{pop} - population size
N_{mate} - mating pool size
P_{mut} - probability of mutating a single parameter

$P_1 = \text{InitPopulation}()$
$\text{Evaluate}(P_1)$
for $i = 1 \to N_{Gen}$ **do**
  $P_i' = \text{SelectMatingPool}(P_i, N_{mate})$
  $P_i' = \text{Crossover}(P_i')$
  $P_i' = \text{Mutate}(P_i', P_{mut})$
  $\text{Evaluate}(P_i')$
  $P_i = P_i \cup P_i'$
  $P_{i+1} = \text{Reduce}(P_i, N_{pop})$
end for

Parameter	Symbol	Value
Number of generations	N_{gen}	30
Size of the population	N_{pop}	100
Size of the mating pool	N_{mate}	50
Number of the best specimens copied to the next population	N_{elite}	20
Probability of mutating each parameter	P_{mut}	0.1

Evolutionary Optimization (3/3)

- **`SelectMatingPool()`**
  - N_{mate} specimens are selected
  - binary tournament
- **`Evaluate()`**
  - intra-day return
  - at least one transaction required
- **`Reduce()`**
  - reduction of parent and offspring population back to N_{pop} specimens
  - N_{elite} best specimens guaranteed to be copied
  - the rest filled using fitness-proportionate selection

Frequent rule set discovery (1/5)

- Frequent rule set
  - we look for trading rules that are „frequently used" by the trading experts
  - a rule *k* is considered „used" by the expert if

$$| \, weight_k \, | \geq minWeight$$

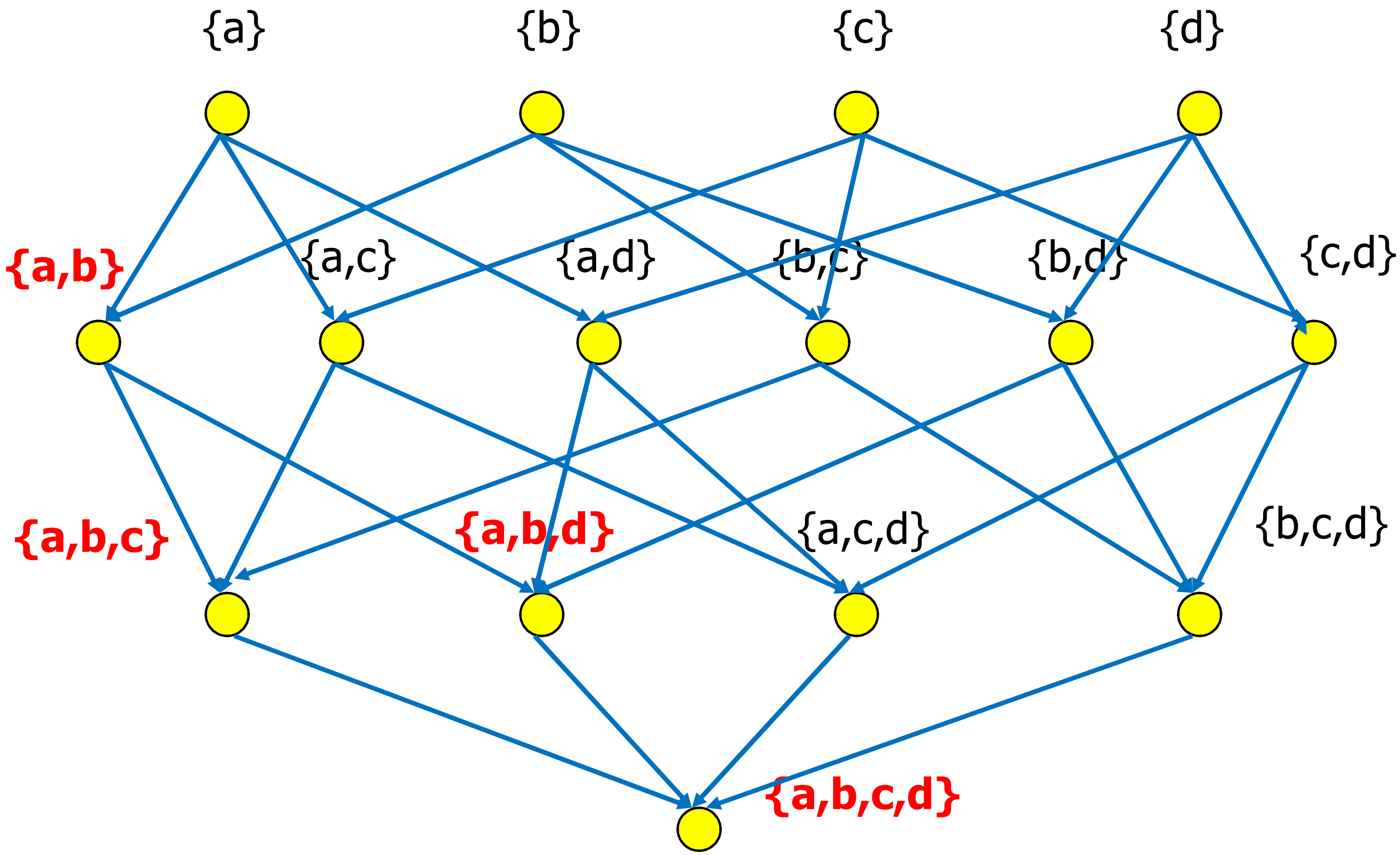
  - a rule is „frequent" if it appears in at least *minSupport* percent of experts

Frequent rule set discovery (2/5)

- Incremental rule set construction
  - similar to frequent set discovery techniques used in association rule mining
  - combinatorial explosion: 2^N possible subsets
  - observation: if a set is frequent then all of its subsets must also be frequent

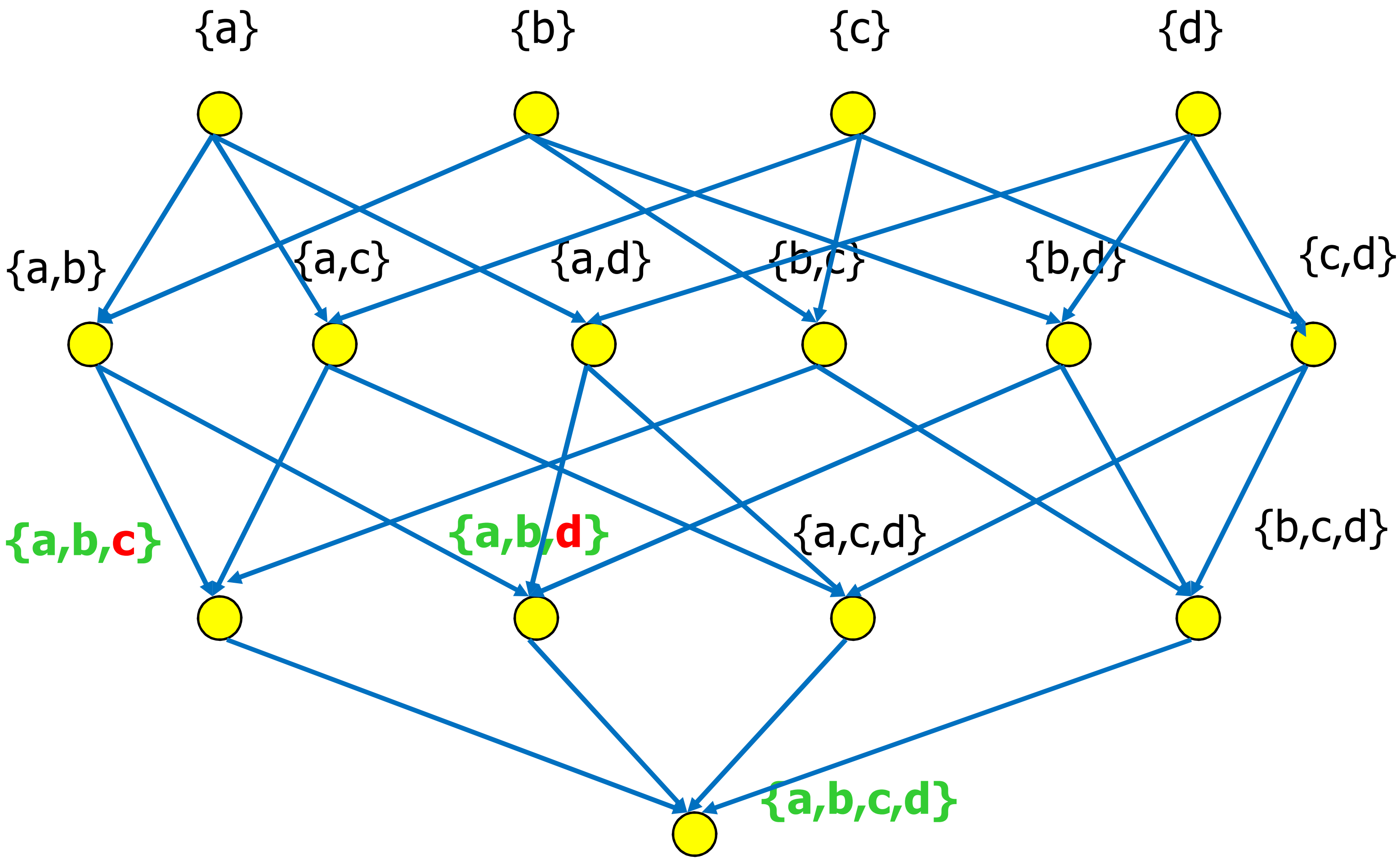
Frequent rule set discovery (3/5)

- If a set is **not** frequent (e.g. **{a, b}**) then all of its supersets can be omitted



Frequent rule set discovery (4/5)

- For each frequent set there exist two frequent subsets differing by only one element



Frequent rule set discovery (5/5)

- Incremental rule set construction (cont.)
  - frequent rule sets can be built incrementally
  - candidate sets of cardinality $N + 1$ can easily be derived from frequent sets of cardinality N
  - start with individual frequent rules
  - stop, if there is no more than one frequent set of cardinality N

Analysis of Trading Rule Sets (1/3)

- Data set for pattern discovery
  - a top 20% of specimens present in the last population in the evolutionary algorithm (after N_{gen} generations)
  - 20% of specimens selected separately for each stock and each trading day
  - all collected specimens used as a single data set

Analysis of Trading Rule Sets (2/3)

Algorithm 2. Frequent rule set discovery algorithm

$i = 1$
C_1 = family of sets, each containing one rule
$S_1 = \text{CheckSupport}(C_1)$
while $|S_i| > 1$ **do**
  $C_{i+1} = \text{Combine}(S_i)$
  $S_{i+1} = \text{CheckSupport}(C_{i+1})$
  $i = i + 1$
end while

minWeight = 0.7

minSupport = 50%

Algorithm 3. CheckSupport - a procedure for selecting rule sets with support $\geq minSupport$

C_i - a family of candidate sets
S_i - a family of frequent sets
V - data set of the best specimens generated by evolutionary algorithm
N - the size of the data set V $(N = |V|)$

$S_i = \emptyset$
for $R \in C_i$ **do**
  $m = 0$
  for $v \in V$ **do**
    if $\forall k \in R \cdot |v[k]| \geq minWeight$ **then**
      $m = m + 1$
    end if
  end for
  if $m \geq minSupport * N$ **then**
    $S_i = S_i \cup \{R\}$
  end if
end for

Analysis of Trading Rule Sets (3/3)

Algorithm 4. Combine - a procedure generating a family C_{i+1} of candidate sets of size $i+1$ based on a family S_i of frequent sets of size i

C_{i+1} - a family of candidate sets of size $i+1$
S_i - a family of frequent sets of size i

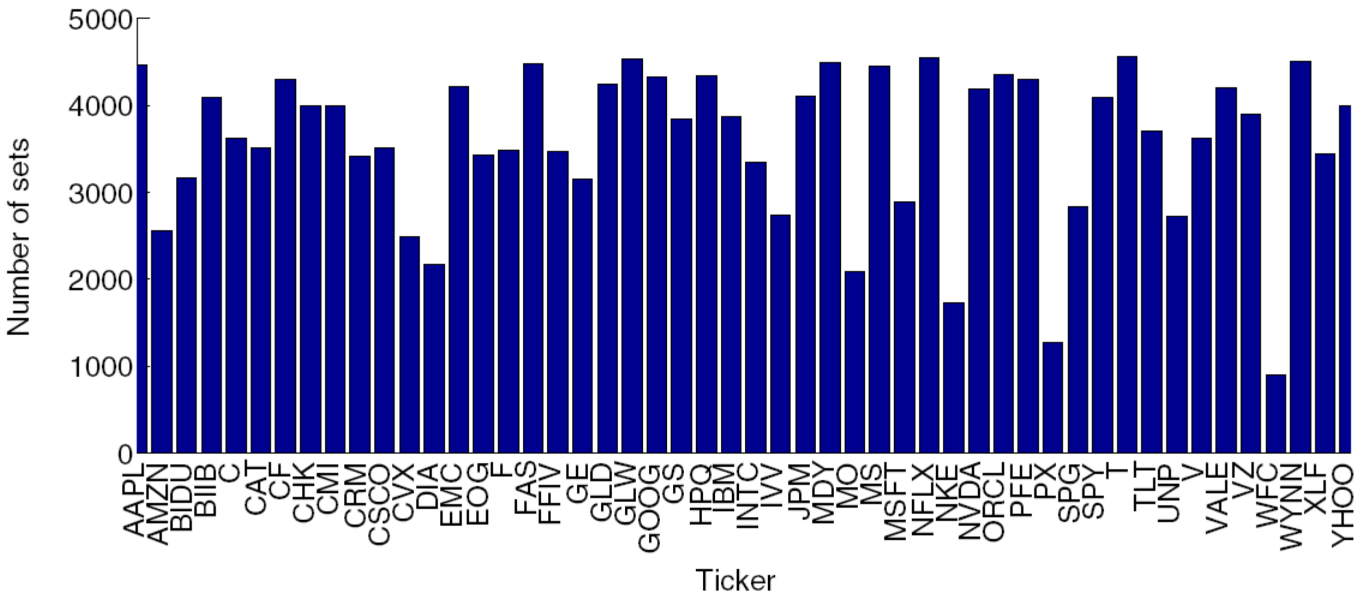
$C_{i+1} = \emptyset$
for $T_j \in S_i$ **do**
  for $T_k \in S_i$ **do**
    if $|T_j \backslash T_k| = 1$ **and** $|T_k \backslash T_j| = 1$ **then**
      $C_{i+1} = C_{i+1} \cup \{T_J \cup T_k\}$
    end if
  end for
end for

Experiments

- High-frequency data from the NYSE
- 50 stocks
- Minute quotations
- Date interval 2012.01.01 to 2012.06.30
- 125 trading days
- 78 rules (96 considering that both EMA and SMA were used)

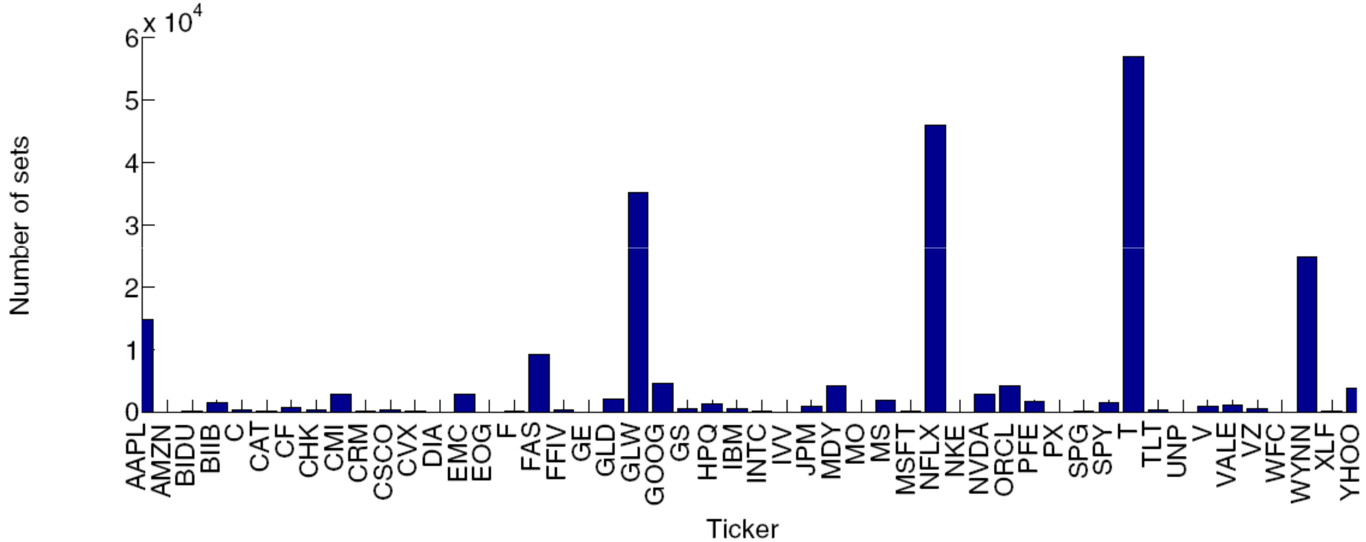
Results (1/5)

- Frequent rule sets of size 2 to 5 found
- Some of the 6-element sets have reached the support of about 47.5%
- 2-element sets



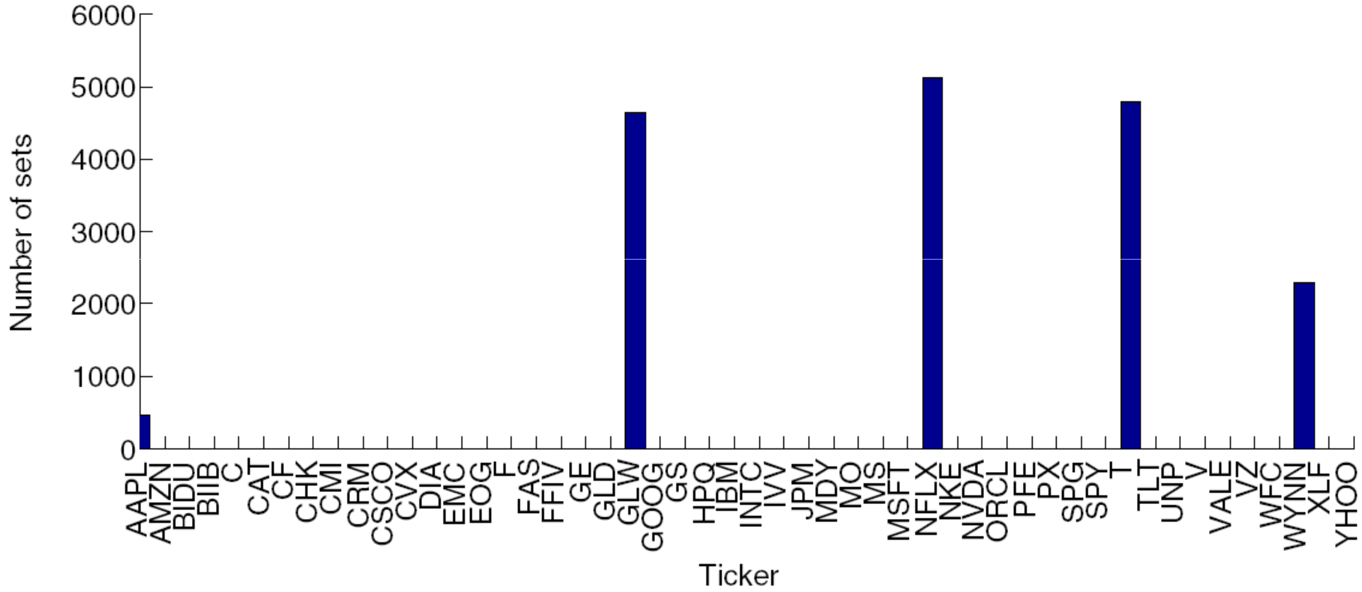
Results (2/5)

- 3-element sets



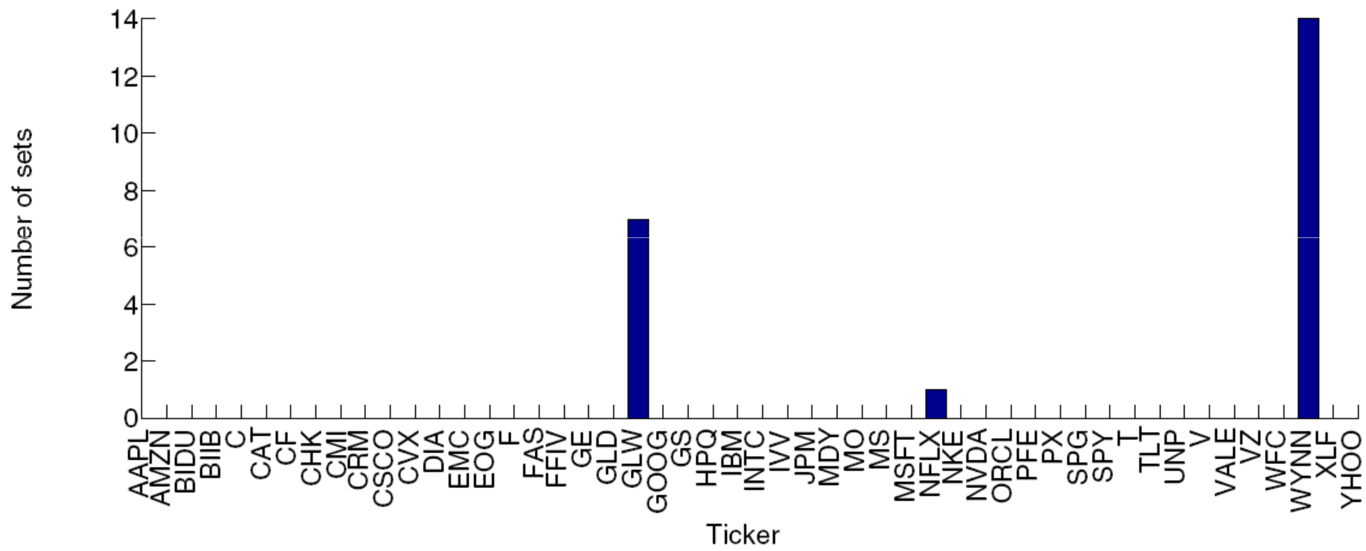
Results (3/5)

- 4-element sets



Results (4/5)

- 5-element sets



Results (5/5)

- Examples of frequent rule sets of size 5 obtained in the experiments (*minSupport* = 50%)

Ticker	Rules
GLW	$EMV_{(3)}$+TBR, $EMV_{(3)}$+Trend, $EMV_{(3)}$+WiO LinDir, $EMV_{(3)}$+NormPrice
NFLX	$EMV_{(2)}$+RoC, K-Stochastic, NormPrice+WiO $EMV_{(2)}$+NormPrice, RSI
T	NormPrice+MA, NormPrice+two MAs, NormPrice+TBR, NormPrice+$V\&P_{(1)}$, $V\&P_{(2)}$
WYNN	$EMV_{(1)}$+OvO, $EMV_{(2)}$+$V\&P_{(2)}$, NormPrice+two MAs, RoC, WiO

Frequent trading rule – example

NormPrice + two MAs

```
if (NormalizedPrice(t) > NormalizedPrice(t - 1))
{
   if (close(t) > MAfast(t))
   {
        if (MAfast(t) > MAslow(t))
        {
                "buy"
        }
   }
}
else
{
   if (close(t) < MAfast(t))
   {
        if (MAfast(t) < MAslow(t))
        {
                "sell"
        }
   }
}
```

Conclusions & Future work (1/2)

- Attempt to find repetitive patterns in trading experts
  - Trading rules using technical indicators
  - Trading experts based on trading rules
  - Parameters of the experts optimized using the EA

- Frequent patterns of up to 5 trading rules emerge in the population

Conclusions & Future work (2/2)

- Repetitive patterns may provide useful information
  - solution improvement
  - quicker generation of good populations for the EA

- Possible extensions
  - Multi-level rule grouping
  - Multiobjective optimization
    - return
    - various risk measures (variance, VaR)
    - other criteria (e.g. similarity to an investment fund[1])

[1] K. Michalak, P. Filipiak, P. Lipiński **"Evolutionary Approach to Multiobjective Optimization of Portfolios That Reflect the Behaviour of Investment Funds"** *Artificial Intelligence: Methodology, Systems, and Applications, Lecture Notes in Computer Science, volume 7557, pp.* 202-211, Springer, 2012.