Automatically Generating Game Tactics through Evolutionary Learning

Marc Ponsen, Héctor Muñoz-Avila,
Pieter Spronck, and David W. Aha

Presented by: Eric Stevens

AI Magazine Volume 27 Number 3 (2006) (© AAAI)

Overview

- **Game AI**
- Real Time Strategy Games
- Dynamic Scripting in Wargus
- Evolutionary State-Based Tactics Generator
- Experimental Results
- Conclusions

Game AI

- Refers to the decision-making process of computer controlled opponents.
- Can help to increase realism of a game.
- Both game industry practitioners and academics predict an increasing importance of game AI.
- High -quality game AI can increase the challenge of the game, and can be a potential selling point

Game AI Continued

- Currently the time spent on AI development is typically short.
- Graphics, and storytelling are the highest priorities.
- AI usually gets developed at the end of the process.
- It is common for even state-of-the-art games to have inferior AI

Adaptive Game AI

- Concerns methods for automatically adapting the behavior of computer controlled opponents
- Dynamic scripting is a reinforcement learning technique used to implement adaptive game AI

Dynamic Scripting

- Employs extensive domain knowledge in the form of a knowledge bases containing tactics
- Manually designing knowledge bases can be very time intensive, and runs the risk of errors in analysis and encoding.
- The authors introduce an automatic way to form these knowledge bases.

Evolutionary State-Based Tactics Generator (ESTG)

- Uses an evolutionary algorithm to generate tactics to be used by dynamic scripting automatically
- The empirical results show that dynamic scripting equipped with the evolved tactics can successfully adapt to static opponents.

Overview

- Game AI
- **Real Time Strategy Games**
- Dynamic Scripting in Wargus
- Evolutionary State-Based Tactics Generator
- Experimental Results
- Conclusions

Real Time Strategy (RTS) Games

- Real Time Strategy (RTS) is a category of strategy games that focus on military combat.
- RTS games require the player to control armies to defeat all opposing forces that are situated on the virtual battlefield in real time.
- The key to winning usually lies in efficiently collecting and managing resources and appropriately allocating these resources over the various action elements.

AI in RTS Games

- Determine all decision for a computer opponent over the course of the entire game.
- Typically the strategy is encoded in the form of scripts
- Scripts list actions that are executed sequentially
- Typical actions include constructing buildings, researching new technologies, and combat

RTS AI Difficulties

- Only partially observable environments
- Adversaries that modify the state asynchronously, and who's decision models are unknown
- Usually an enormous number of possible actions that can be executed at any given time, and some of their effects on the state are uncertain.
- Decisions must be made under time constraints

Overview

- Game AI
- Real Time Strategy Games
- **Dynamic Scripting in Wargus**
- Evolutionary State-Based Tactics Generator
- Experimental Results
- Conclusions

Definitions

- Action: An atomic transformation in the game situation.
- Tactic: a sequence consisting of one or more primitive actions.
- Strategy: A sequence of tactics that can be used to play a complete game.

Reinforcement Learning with Dynamic Scripting

- Reinforcement Learning: an adaptive agent interacts with its environment and learns what to do, and when to achieve a certain goal based on a scalar reward signal that it receives from the environment
- Has been designed so that adaptive agents only exploit knowledge in a few trials, allowing a balance between exploitation and exploration by maintaining a minimum and maximum selection probability for all actions.

Reinforcement Learning with Dynamic Scripting Continued...

- All state-action values are updated in a specific state through a redistribution process so that the sum of the state-action values remains constant.
- The goal is to avoid an “Optimal Policy” since this may result in overfitting to one specific strategy.
- Dynamic scripting is capable of generating a variety of behaviors and responding quickly to changing game dynamics

Wargus

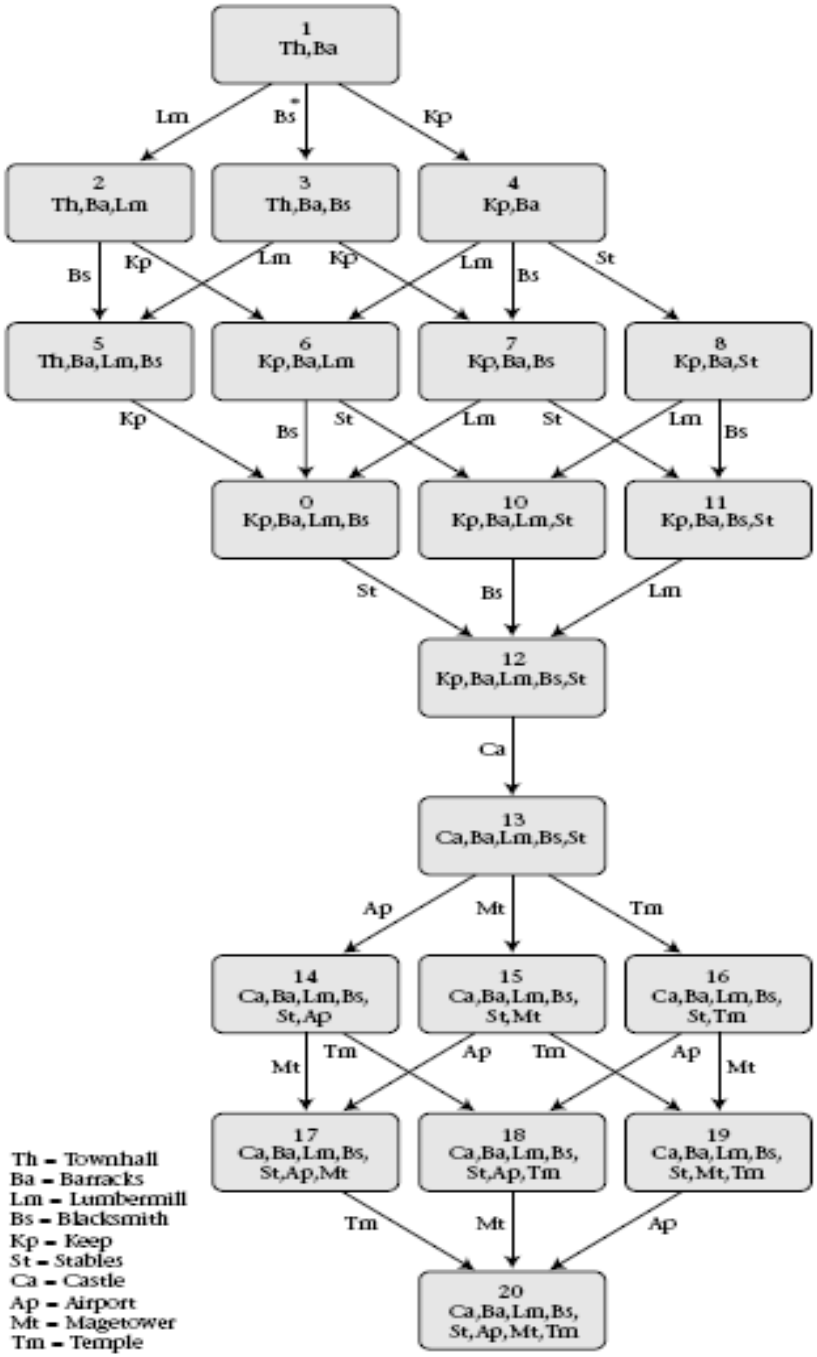
- The experiments were done on the RTS game Wargus.
- Wargus is a clone of the popular game Warcaftt II

Dynamic Scripting in Wargus

- An agent controlled by Dynamic Scripting, called the adaptive agent, plays against a static agent.
- The static agent executes a static script representing a strategy.
- The adaptive agent generates scrips on the fly bases on its current policy.

States and Their Knowledge Bases

- The authors divide the Wargus game into a small number of abstract states, each corresponding to a unique knowledge base, whose tactics can be selected by dynamic scripting when the game is in that particular state
- States were distinguished according to the types of available buildings
- State changes are spawned by tactics that create new buildings


1
Th,Ba
Lm
Bs
Kp
2
Th,Ba,Lm
3
Th,Ba,Bs
4
Kp,Ba
Bs
Kp
Lm
Kp
Lm
Bs
St
5
Th,Ba,Lm,Bs
6
Kp,Ba,Lm
7
Kp,Ba,Bs
8
Kp,Ba,St
Kp
Bs
St
Lm
St
Lm
Bs
9
Kp,Ba,Lm,Bs
10
Kp,Ba,Lm,St
11
Kp,Ba,Bs,St
St
Bs
Lm
12
Kp,Ba,Lm,Bs,St
Ca
13
Ca,Ba,Lm,Bs,St
Ap
Mt
Tm
14
Ca,Ba,Lm,Bs,
St,Ap
15
Ca,Ba,Lm,Bs,
St,Mt
16
Ca,Ba,Lm,Bs,
St,Tm
Mt
Tm
Ap
Tm
Ap
Mt
17
Ca,Ba,Lm,Bs,
St,Ap,Mt
18
Ca,Ba,Lm,Bs,
St,Ap,Tm
19
Ca,Ba,Lm,Bs,
St,Mt,Tm
Tm
Mt
Ap
20
Ca,Ba,Lm,Bs,
St,Ap,Mt,Tm
Th = Townhall
Ba = Barracks
Lm = Lumbermill
Bs = Blacksmith
Kp = Keep
St = Stables
Ca = Castle
Ap = Airport
Mt = Magetower
Tm = Temple

Process for Dynamic Scripting

- Selecting tactics for the first state.
- Upon selecting a tactic, a state change occurs, and then a new tactic is selected for that state.
- To avoid monotonous behavior, each tactic is restricted to be selected only once per state.
- Tactics are selected until either N (N= 100) tactics are selected, or until the final state 20 is reached.
- A maximum of M tactics must be selected (m= 20) for the final state.
- The script then moves into an attack loop.

Weight Value Adaptation

- Dynamic scripting maintains an associated weight value that indicates the desirability of choosing that tactic in the specific state.
- All weights were initialized to 100 at the start of the experiments.
- After each game, the weight values of all the tactics employed are updated.
- The magnitude of the weight adjustments in a state is uniformly distributed over the nonselected tactics for that state.

Weight Value Adaptation Continued

- The size of the weight value updates are determined mainly by a state reward.
- To recognize the importance of winning a game, weight value updates also take into account a global reward.

State Reward Function

- The state reward function *R* for state *i*, for the adaptive agent *a* yields a value in the range [0, 1]

$$R_i = \frac{(S_{a,i} - S_{a,i-1})}{(S_{a,i} - S_{a,i-1}) + (S_{s,i} - S_{s,i-1})}$$

- $S_{a,x}$ represents the score of the adaptive agent *a* after state *x*, $S_{s,x}$ represents the score of the static agent *s* after state *x*,
- The score is a value that measures the success of an agent up to the moment the score is calculated. The score never decreases during game play.

Global Reward Function

- The global reward function R fro the adaptive agent a yields a value in the range [0, 1]

$$R_\infty = \begin{cases} \min\left(\frac{S_{a,L}}{S_{a,L} + S_{s,L}}, b\right) & \text{if } a \text{ lost,} \\ \max\left(\frac{S_{a,L}}{S_{a,L} + S_{s,L}}, b\right) & \text{if } a \text{ won.} \end{cases}$$

- L is the number of the state that the game ended in.
- b $\in$ (0, 1) and is the break even point.

Score Function

- The score $S_{x,y}$ for agent x, after state y is defined by

$$S_{x,y} = C\,M_{x,y} + (1 - C)\,B_{x,y}$$

- $M_{x,y}$ represents the military points scored
- $B_{x,y}$ represents the building points scored
- The constant C represents the weight given to Military points, and is set to .7

Weight Value Adjustment

- Weight values are bounded by $[w_{min}, w_{max}]$
- A new weight is calculated as w + delta w

$$\Delta W = \begin{cases} -P_{max}\left(C_{end}\ \frac{b-R_{\infty}}{b} + (1\ -C_{end})\ \frac{b-R_i}{b}\right) \\ R_{max}\left(C_{end}\ \frac{R_{\infty}-b}{1-b} + (1\ -C_{end})\ \frac{R_i-b}{1-b}\right) \end{cases}$$

- R_{max} and P_{max} are the maximum reward, and maximum penalty
- R is the global reward function
- R_i is the state reward for the state corresponding to the knowledge base containing the weight value.

Experiment Values

- P_{max} and R_{max} =400
- W_{max} =4000, W_{min} =25
- b =.5
- C_{end} = .3 . This variable represents the fraction of the weight value adjustment that is determined by the global function reward.

$$\Delta W = \begin{cases} -P_{max}\left(C_{end}\ \frac{b-R_{\infty}}{b} + (1\ -C_{end})\ \frac{b-R_i}{b}\right) \\ R_{max}\left(C_{end}\ \frac{R_{\infty}-b}{1-b} + (1\ -C_{end})\ \frac{R_i-b}{1-b}\right) \end{cases}$$

Overview

- Game AI
- Real Time Strategy Games
- Dynamic Scripting in Wargus
- **Evolutionary State-Based Tactics Generator**
- Experimental Results
- Conclusions

Automatically Generating Tactics

- Step 1 Evolutionary Algorithm (EA):
  - An evolutionary algorithm is used to search for strategies that defeat specific opponent strategies
  - The training set consist of 40 different strategies, 36 which are evolutionary scripts previously evolved.
- Step 2 Knowledge Transfer (KT)
  - A state-based knowledge transfer from evolved strategies to tactics
- Evaluation
  - Test the evolved tactics with dynamic scripting

Evolutionary Algorithm

- A population of chromosomes, each of which represents a static strategy.
- A chromosome is divided into 20 states
- States have a list of genes
- Each gene represents a game action
- Gene Types:
  - Build Genes
  - Research Genes
  - Economy Genes
  - Combat Genes

Evolutionary Algorithm Continued...

- Chromosome for the initial population are generated randomly.
- To determine the fitness of a chromosome, it is translated to a game AI script, and played against a script in the training set.
- The fitness function F for the adaptive agent a yields a value in the range, [0,1]

Fitness Function

$$F = \begin{cases} \min\left(\frac{C_T}{C_{max}} \cdot \frac{M_a}{M_a + M_s}, b\right) & \text{if } a \text{ lost,} \\ \max\left(\frac{M_a}{M_a + M_s}, b\right) & \text{if } a \text{ won.} \end{cases}$$

C_T represents the time step at which the game was finished.

- C_{max} is the max time steps a game can be played
- M_a represents the military points for the adaptive agent.
- M_s represents the military points for the opponent.

Evolutionary Algorithm Continued...

- The authors goal is to create a chromosome with a fitness exceeding a target value (.7)
- Evolution ends once such a chromosome is found.
- If no such chromosome is found, evolution also ends after a maximum number of chromosomes are created (250).
- Only relatively successful chromosomes are aloud to breed.

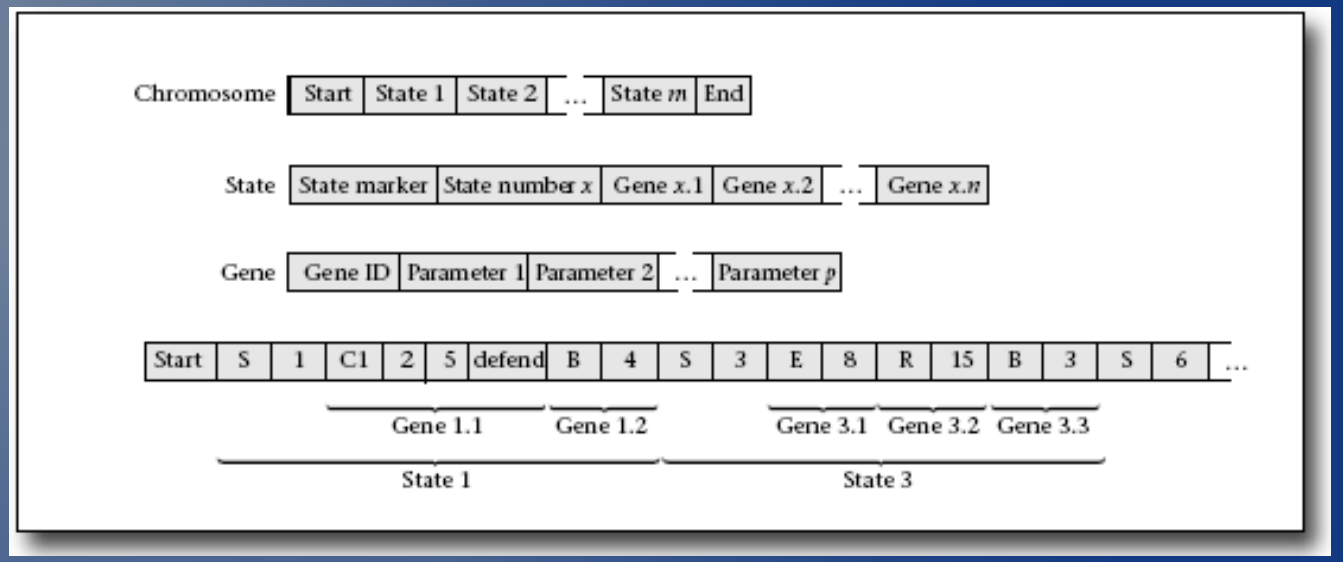
Genetic Operators

- State Crossover (30% chance) – Selects two parents and copies states from either parent to the child chromosome.
- Gene Replace Mutation (30%) – Selects one parent, and replaces build, economy, research, or combat genes with a 25 percent probability
- Gene Biased Mutation (30%) – Selects one parent, and mutates parameters for existing economy or combat genes with a 50 percent probability
- Randomization (10%) – Randomly generates a new Chromosome.

State Based Knowledge Transfer

- ESTG automatically recognizes and extracts tactics from the evolved chromosomes and inserts these into state-specific knowledge bases.
- All genes grouped in an activated state in the chromosome are considered to be a single tactic.

State Based Knowledge Transfer Continued...



- First Tactic
  - includes genes 1.1 and 1.2
  - Will be inserted into the knowledge base for state 1
- Second Tactic
  - Since gene 1.2 spawns a state change, the next genes will be part of a tactic for state 3

Overview

- Game AI
- Real Time Strategy Games
- Dynamic Scripting in Wargus
- Evolutionary State-Based Tactics Generator
- **Experimental Results**
- Conclusions

Experimental Evaluation

- 40 chromosomes were evolved against the strategies provided in the training set.
- All had a fitness score higher than .7 which represents a clear victor.
- In the KT step, the 40 evolved chromosomes produced 164 tactics that were added to the evolved knowledge bases for their corresponding state.
- No tactics were found for some later states due to the fact that all games ended before the adaptive agent constructed all buildings.

Performance of Dynamic Scripting

- A sequence of 100 games constituted one experiment, and the adaptive agent's policy was updated after each game.
- 10 experiments were ran, each against four different strategies for the static agent.
- Small Balanced Land Attack (SBLA)
- Large Balanced Land Attack (LBLA)
- Soldier's Rush (SA)
- Knight's Rush (KR)

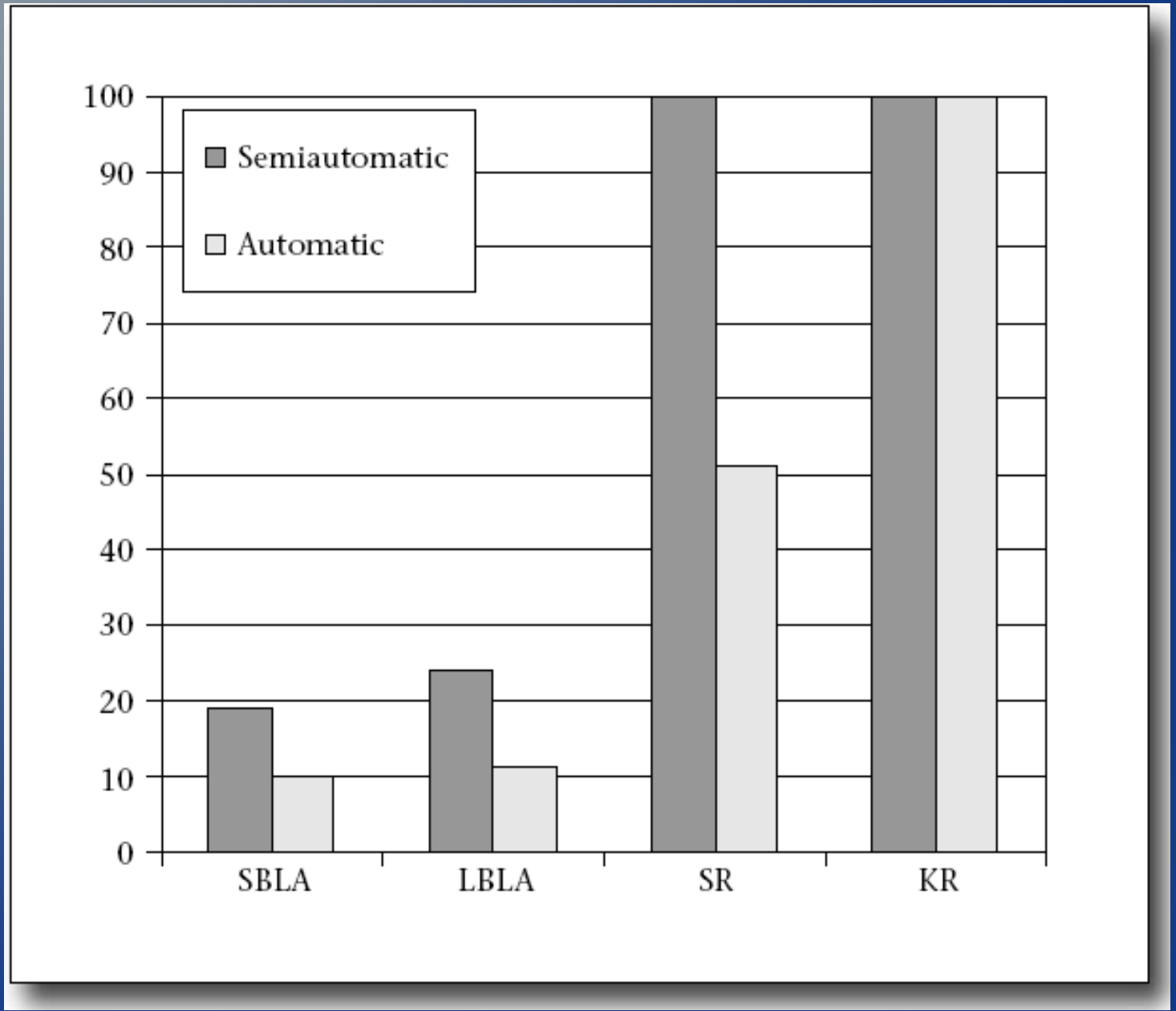
Performance of Dynamic Scripting Continued...

- To quantify the performance of adaptive agent against the static agent, the randomization turning point (RTP) was used.
- After each game a randomization test was performed using the global reward values over the last 10 games with the null hypotheses that both agents are equally strong.
- The adaptive agent outperformed the static agent if the randomization test concluded that the null hypotheses can be rejected 90% of the time.
- The RTP is the number of the first game that the adaptive agent outperforms the static agent.

Semiautomatic Approach

- Ponsen and Spronck (2004) manually improved existing knowledge bases from counter strategies that were evolved offline and tested dynamic scripting against SBLA, LBLA, SR, and KR.

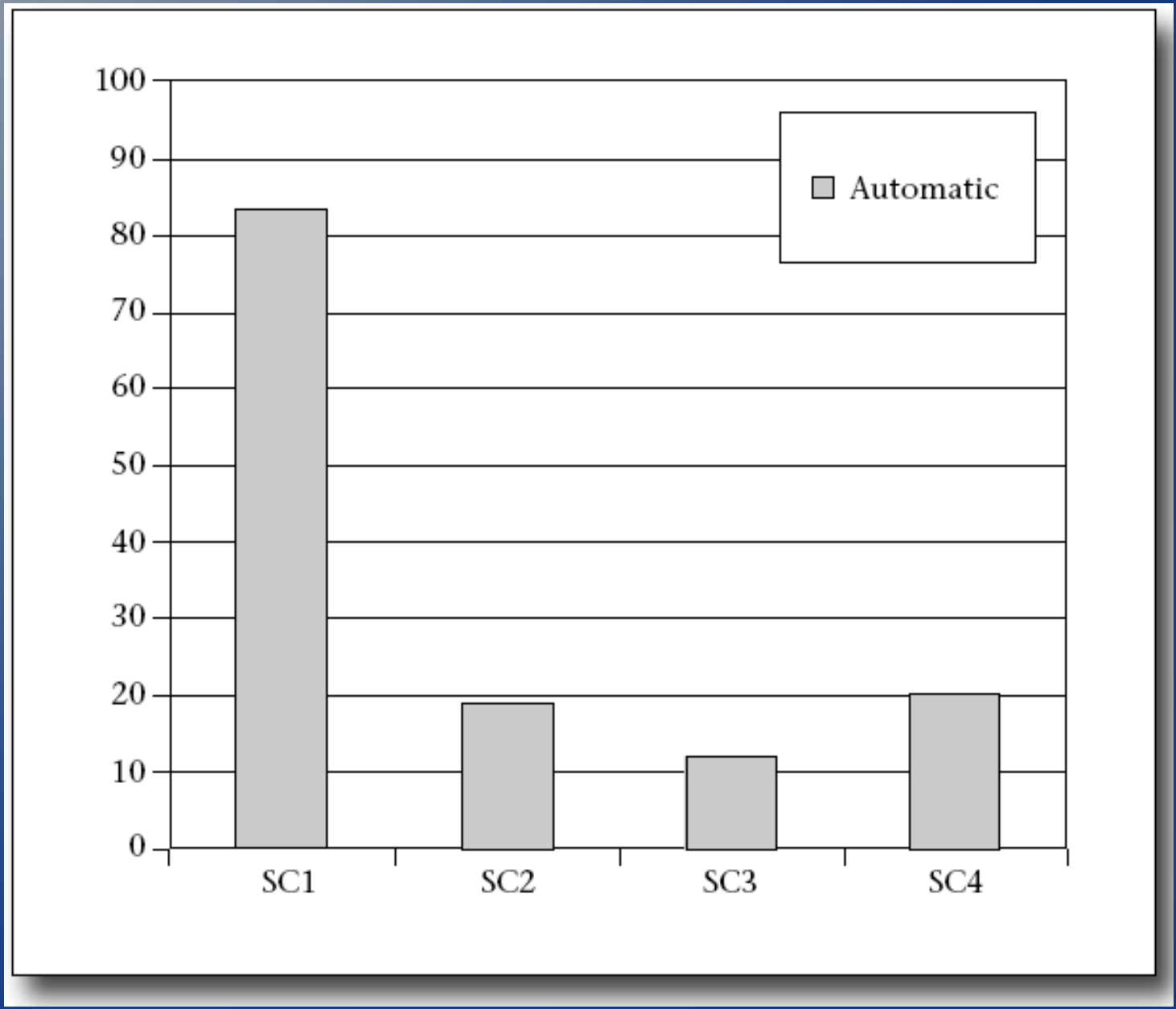
Automatic vs Semiautomatic



Performance Increase

- The evolved knowledge bases were not restricted to the domain knowledge provided by the designer
  - They were both created and improved manually in the Semiautomatic experiments.
- The automatically generated knowledge bases included tactics that consist of multiple primitive actions, whereas the knowledge bases used in earlier experiments mostly include tactics that consist for a single primitive action.

Student Created Scripts



Overview

- Game AI
- Real Time Strategy Games
- Dynamic Scripting in Wargus
- Evolutionary State-Based Tactics Generator
- Experimental Results
- **Conclusions**

Conclusions

- Using the Evolutionary State-Based Tactics Generator, knowledge bases of state-based tactics can be evolved for dynamic scripting
- The technique was applied to the game Wargus, in an attempt to create an adaptive opponent.
- The results demonstrated that the dynamic scripting using the ESTG evolved knowledge bases can adapt to many different static strategies, even to previously unseen ones.

Questions?