# New Splitting Approach for Adaptative Tile Coding

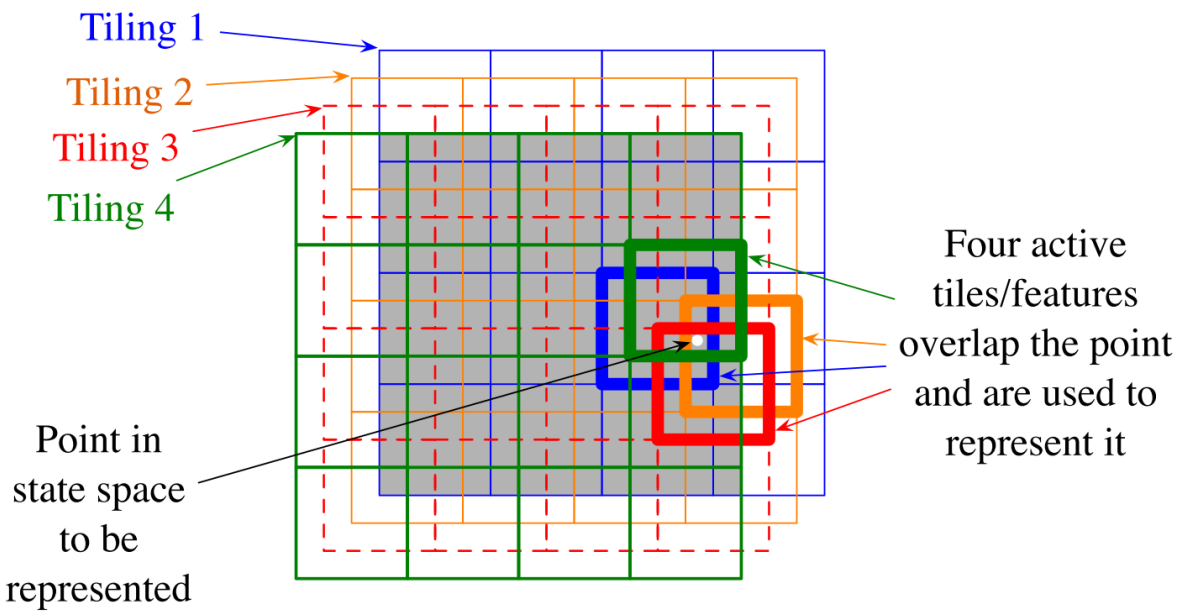
## Course Project – Reinforcement Learning

Mathieu Nassif

McGill University

April 20, 2017

# Problems of Tile Coding



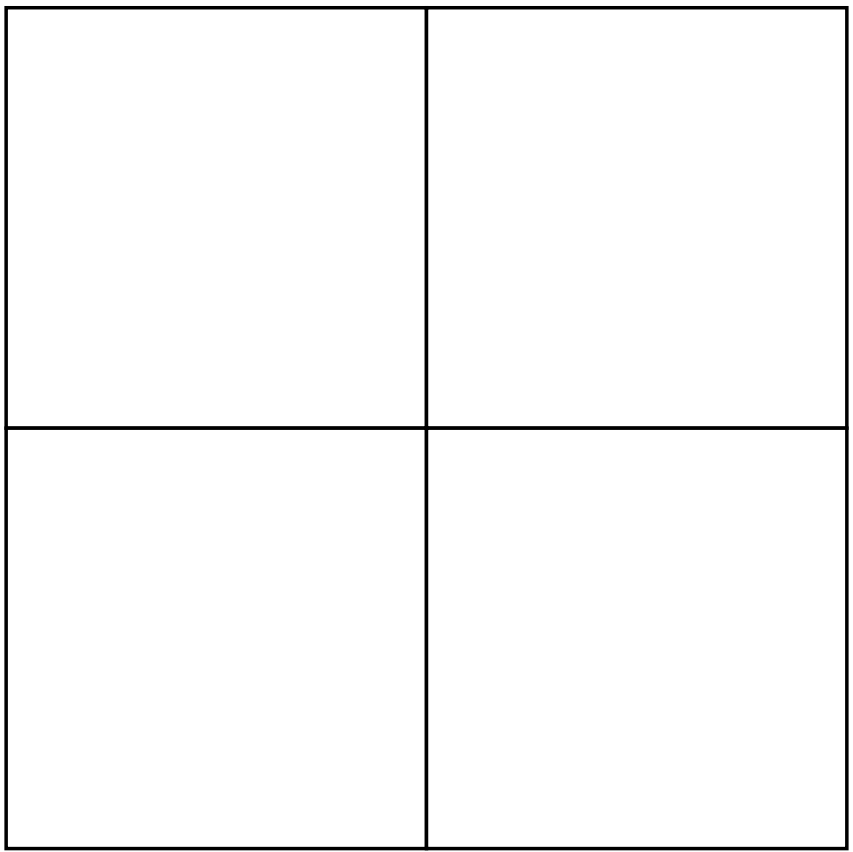

– # of tilings?
– Size of each tile?

Propagation
VS
Generalization

# Adaptative Tile Coding

*S. Whiteson, M. E. Taylor and P. Stone*

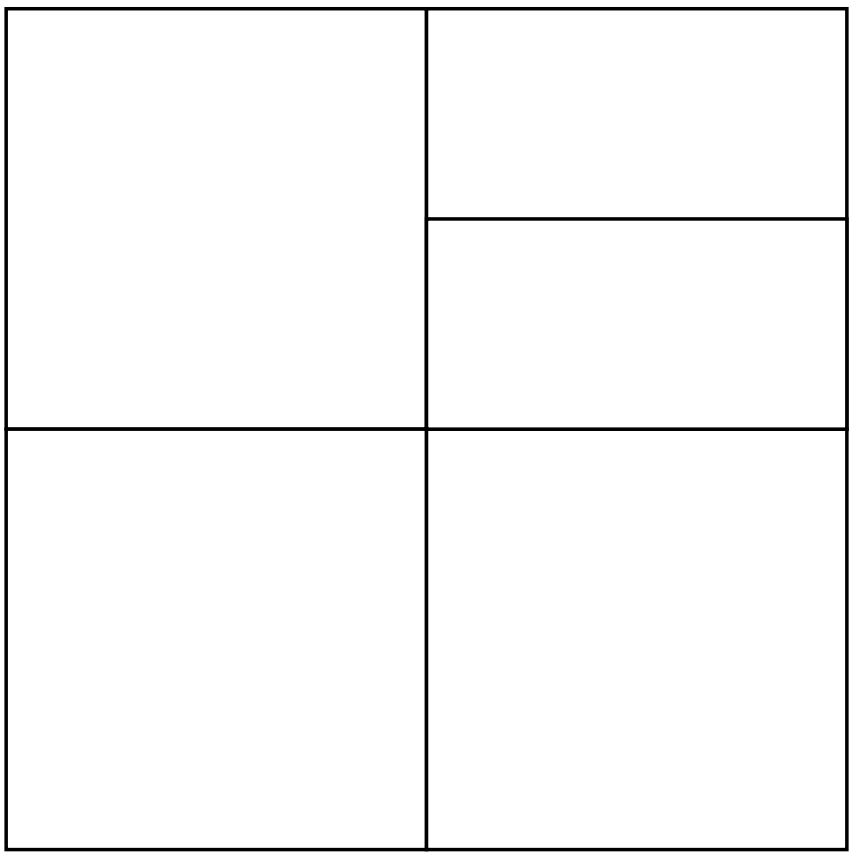
1. Start with **ONE** very coarse tiling



# Adaptative Tile Coding

*S. Whiteson, M. E. Taylor and P. Stone*

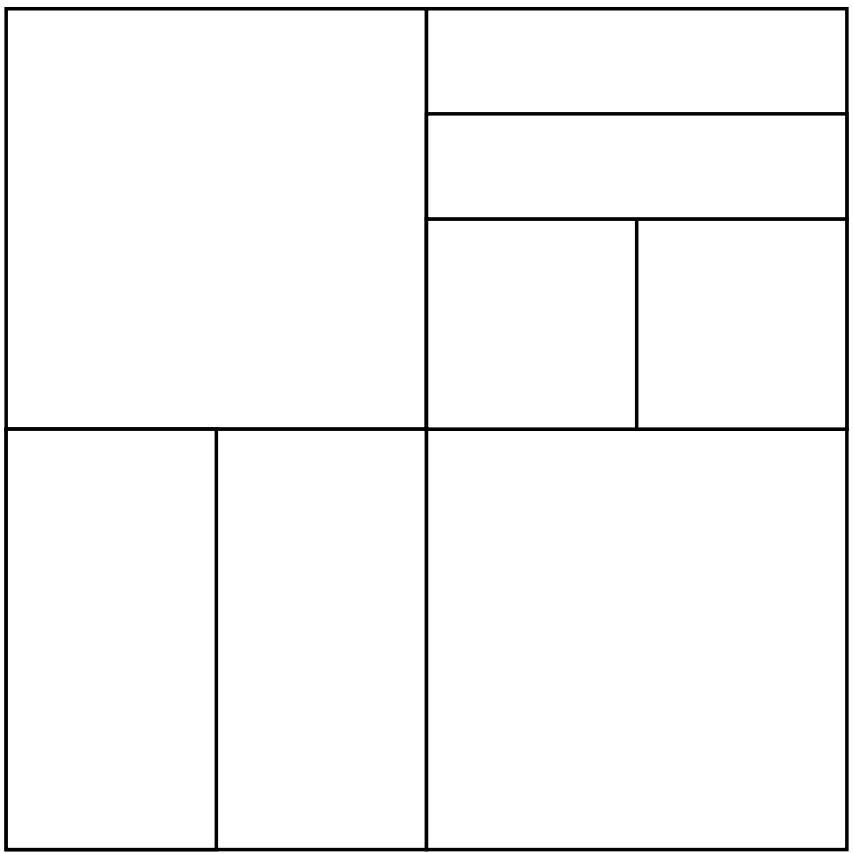
1. Start with **ONE** very coarse tiling
2. When the updates stabilize, choose a tile to split



# Adaptative Tile Coding

*S. Whiteson, M. E. Taylor and P. Stone*

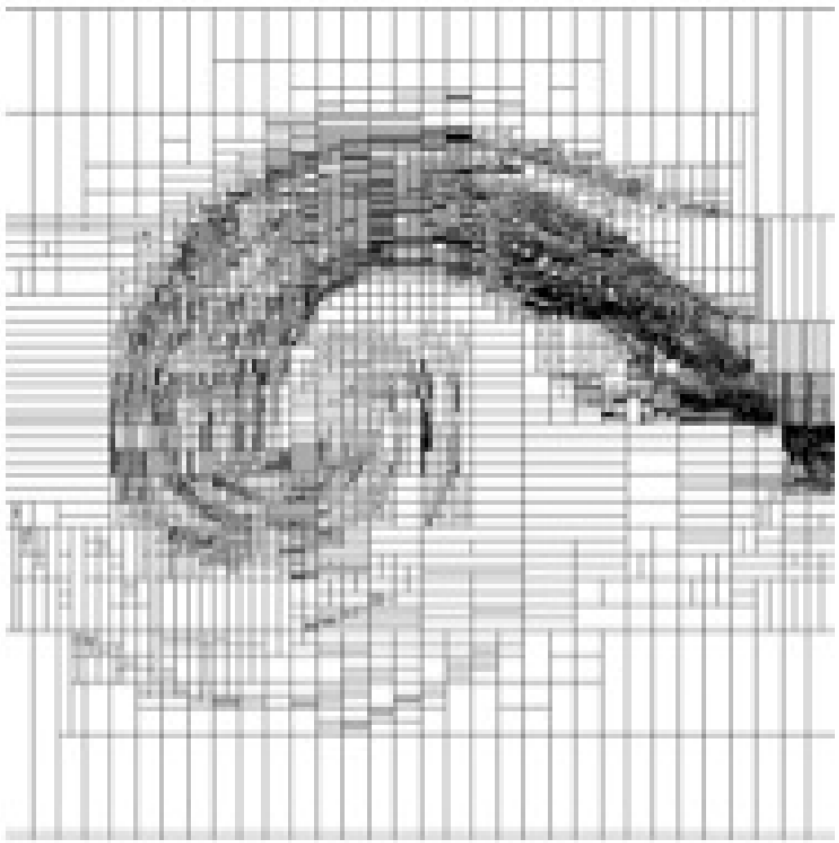
1. Start with **ONE** very coarse tiling
2. When the updates stabilize, choose a tile to split
3. Split the tile along one dimension in half



# Adaptative Tile Coding

*S. Whiteson, M. E. Taylor and P. Stone*

1. Start with **ONE** very coarse tiling
2. When the updates stabilize, choose a tile to split
3. Split the tile along one dimension in half
4. Repeat



# New Splitting Approach

**Objectives**

- Find near-optimal split (not always in half)
- Keep constant memory usage per tile

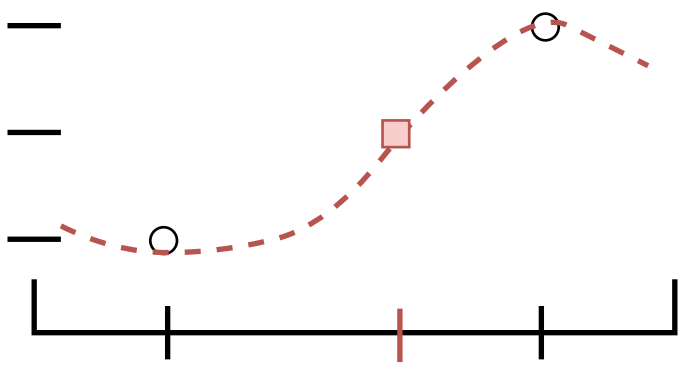
# New Splitting Approach

**Objectives**

- Find near-optimal split (not always in half)
- Keep constant memory usage per tile

**New Strategy**

1. *Analyze for each dimension*
2. Find the maximum and minimum rewards in each tile
3. Find the middle value: 0.5 * (max + min)
4. Estimate the position of the middle value
5. Split at this position



# Estimating the Position of the Middle Point

**Challenge:** Find middle point from a stream of data (no memory)

# Estimating the Position of the Middle Point

**Challenge:** Find middle point from a stream of data (no memory)

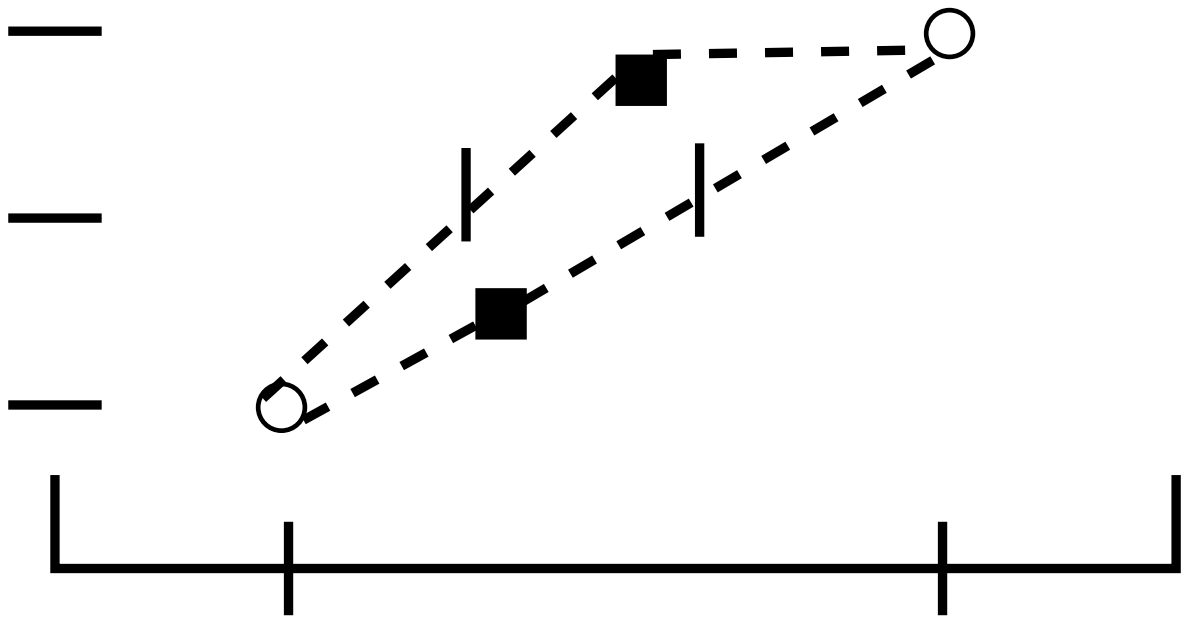
**Extremums:** Keep a variable for the maximum and the minimum

# Estimating the Position of the Middle Point

**Challenge:** Find middle point from a stream of data (no memory)

**Extremums:** Keep a variable for the maximum and the minimum

**Middle Point:** Use linear intrapolations for every new point, and average them.



# Results