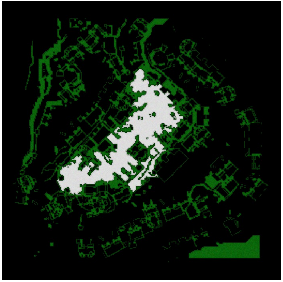
Euclidean Heuristic Optimization

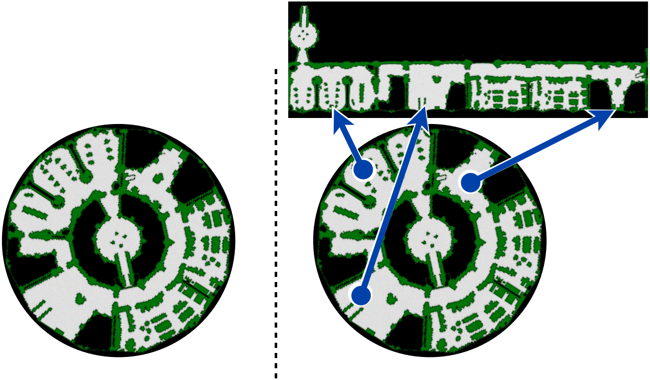
Chris Rayner (University of Alberta)
Michael Bowling (University of Alberta)
Nathan Sturtevant (University of Denver)

AAAI 2011 August 9, 2011



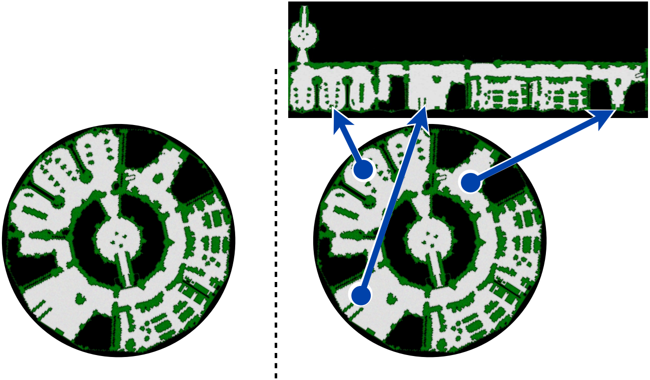
Build memory-efficient data structures *before shipping* to help:

- Solve “easy” problems in rapid succession
- Any state can be the goal



Dragon Age: Origins (BioWare)

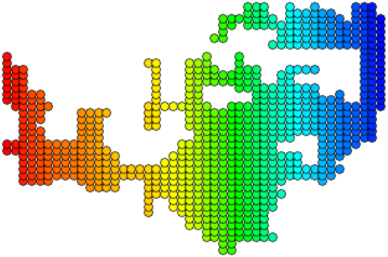
- Straight-line heuristics are easy to compute *online*



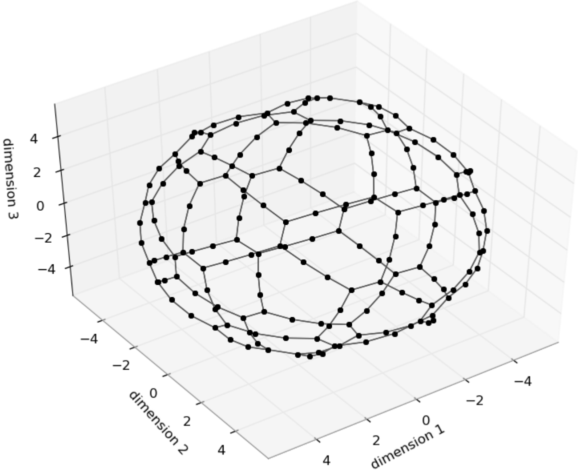
Dragon Age: Origins (BioWare)

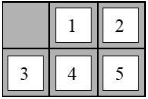
- Straight-line heuristics are easy to compute *online*
- Improve them by (re-)arranging the search graph *offline*

(Example 1) Rearranging a grid world



(Example 2) Arranging a 2×3 sliding tile puzzle:





EUCLIDEAN HEURISTICS

Definition (Euclidean heuristics)

A Euclidean heuristic Y is one whose heuristic values can be computed as distances in a Euclidean space of d dimensions

$$h(i,j) = \|y_i - y_j\|$$

EUCLIDEAN HEURISTICS

Definition (Euclidean heuristics)

A Euclidean heuristic Y is one whose heuristic values can be computed as distances in a Euclidean space of d dimensions

$$h(i, j) = \|y_i - y_j\|$$

Euclidean Heuristics

Definition (Euclidean heuristics)

A Euclidean heuristic Y is one whose heuristic values can be computed as distances in a Euclidean space of d dimensions

$$h(i, j) = \|y_i - y_j\|$$

PROBLEM STATEMENT

Look among *all* Euclidean heuristics Y for one that is best

Problem Statement

Look among *all* Euclidean heuristics Y for one that is best

Definition (Optimal Euclidean Heuristic)

Minimizes the loss between true distances δ and heuristics h:

$$\begin{aligned} \mathop{\textbf{minimize}}_{Y} \quad & \mathcal{L}(Y) \\ \textbf{subject to} \quad & Y \text{ is admissible and consistent} \end{aligned}$$

Problem Statement

Look among *all* Euclidean heuristics $\mathcal{Y}$ for one that is best

Definition (Optimal Euclidean Heuristic)

Minimizes the loss between true distances δ and heuristics h:

$$\begin{array}{ll} \underset{\mathcal{Y}}{\textbf{minimize}} & \mathcal{L}(\mathcal{Y}) \\ \textbf{subject to} & \mathcal{Y} \text{ is admissible and consistent} \end{array}$$

Admissibility/Consistency

Simplified Constraints

A Euclidean heuristic Y is admissible and consistent if:

$$\forall(i,j) \quad \|y_i - y_j\| \leq \delta(i,j)$$
$$\forall(i,j,k) \quad \|y_i - y_j\| \leq \delta(i,k) + \|y_j - y_k\|$$

Admissibility / Consistency

Simplified Constraints

A Euclidean heuristic Y is admissible and consistent if:

$$\forall (i,j) \quad \|y_i - y_j\| \leq \delta(i,j)$$
$$\forall (i,j,k) \quad \|y_i - y_j\| \leq \delta(i,k) + \|y_j - y_k\|$$

Theorem

Y is locally admissible

$$\forall (i,j) \in E \; \|y_i - y_j\| \leq \delta(i,j)$$

$$\Updownarrow$$

Y is admissible and consistent

Passino and Antsaklis, 1994

Admissibility / Consistency

Simplified Constraints

A Euclidean heuristic Y is admissible and consistent if:

$$\forall (i,j) \quad \|y_i - y_j\| \leq \delta(i,j)$$
$$\forall (i,j,k) \quad \|y_i - y_j\| \leq \delta(i,k) + \|y_j - y_k\|$$

Theorem

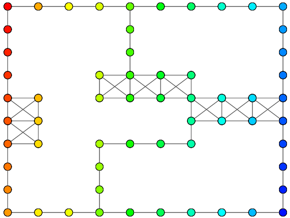
Y is locally admissible

$$\forall (i,j) \in E \ \|y_i - y_j\| \leq \delta(i,j)$$

$$\Updownarrow$$

Y is admissible and consistent

Passino and Antsaklis, 1994



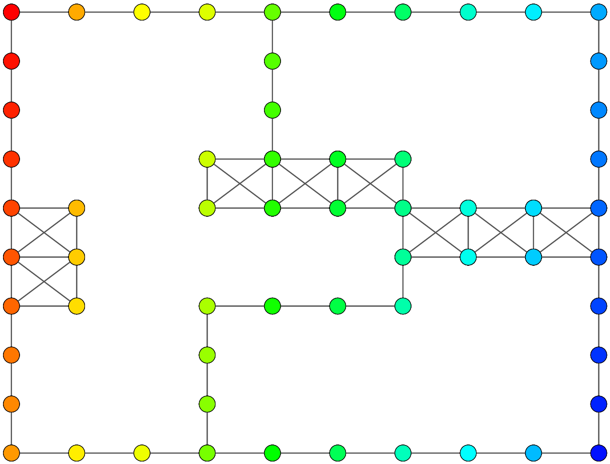
(one constraint per edge)

Definition (Optimal Euclidean Heuristic)

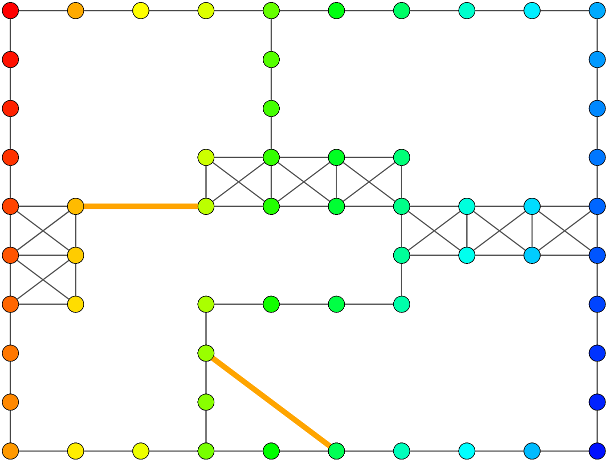
Minimizes the loss between true distances δ and heuristics h:

$$\begin{aligned} \underset{Y}{\textbf{minimize}} \quad & \mathcal{L}(Y) \\ \textbf{subject to} \quad & Y \text{ is admissible and consistent} \end{aligned}$$

Defining loss

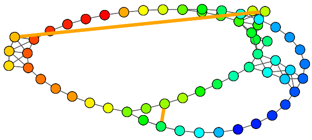


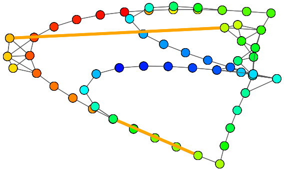
DEFINING LOSS



Defining loss

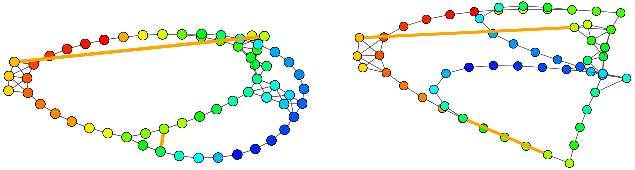
Which is better?





Defining loss

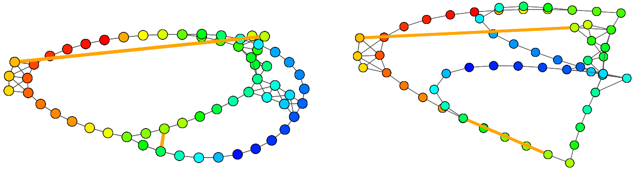
Which is better?



Loss function $\mathcal{L}$ combines errors $\forall(i, j)$ into a single scalar

Defining loss

Which is better?

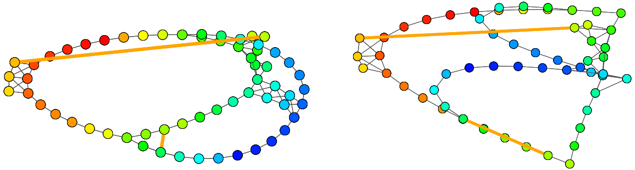


Loss function $\mathcal{L}$ combines errors $\forall(i,j)$ into a single scalar

- specify trade-off: many small errors *vs.* a large error?

DEFINING LOSS

Which is better?

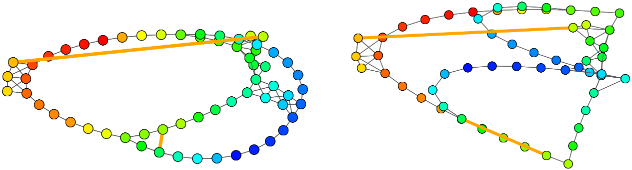


Loss function $\mathcal{L}$ combines errors $\forall(i, j)$ into a single scalar

- specify trade-off: many small errors *vs.* a large error?
- specify relative importance of each state pair

Defining loss

Which is better?



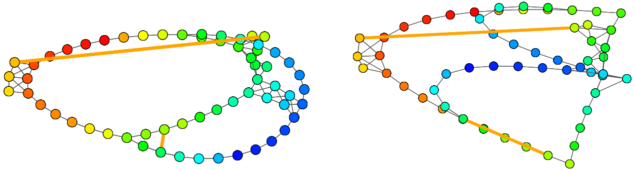
Loss function $\mathcal{L}$ combines errors $\forall(i,j)$ into a single scalar

- specify trade-off: many small errors *vs.* a large error?
- specify relative importance of each state pair

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} \left| \delta(i,j)^2 - \|y_i - y_j\|^2 \right|$$

Defining loss

Which is better?



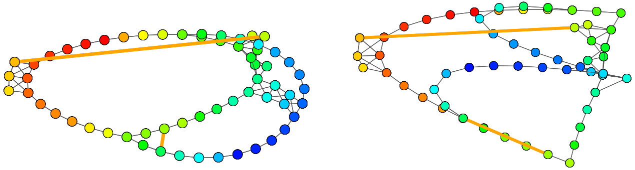
Loss function $\mathcal{L}$ combines errors $\forall(i,j)$ into a single scalar

- specify trade-off: many small errors *vs.* a large error?
- specify relative importance of each state pair

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} \left| \delta(i,j)^2 - \|y_i - y_j\|^2 \right|$$

DEFINING LOSS

Which is better?



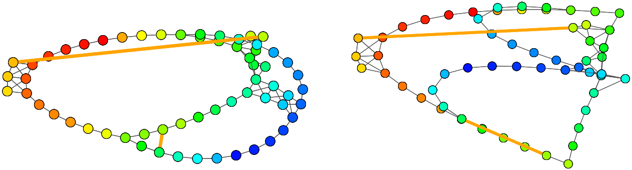
Loss function $\mathcal{L}$ combines errors $\forall(i,j)$ into a single scalar

- specify trade-off: many small errors *vs.* a large error?
- specify relative importance of each state pair

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} \left| \delta(i,j)^2 - \|y_i - y_j\|^2 \right|$$

DEFINING LOSS

Which is better?



Loss function $\mathcal{L}$ combines errors $\forall(i,j)$ into a single scalar

- specify trade-off: many small errors *vs.* a large error?
- specify relative importance of each state pair

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} \left| \delta(i,j)^2 - \|y_i - y_j\|^2 \right|$$

Defining loss

Simplifying the Objective

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} |\delta(i,j)^2 - \|y_i - y_j\|^2|$$

Defining loss

Simplifying the Objective

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} |\delta(i,j)^2 - \|y_i - y_j\|^2|$$

$$\text{Admissibility} \rightarrow \delta(i,j)^2 \geq \|y_i - y_j\|^2$$

Defining loss

Simplifying the Objective

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} |\delta(i,j)^2 - \|y_i - y_j\|^2|$$

$$\text{Admissibility} \rightarrow \delta(i,j)^2 \geq \|y_i - y_j\|^2$$

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} (\delta(i,j)^2 - \|y_i - y_j\|^2)$$

Defining loss

Simplifying the Objective

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} |\delta(i,j)^2 - \|y_i - y_j\|^2|$$

$$\text{Admissibility} \rightarrow \delta(i,j)^2 \geq \|y_i - y_j\|^2$$

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} (\delta(i,j)^2 - \|y_i - y_j\|^2)$$

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} \delta(i,j)^2 - \sum_{i,j} W_{ij} \|y_i - y_j\|^2$$

Defining loss

Simplifying the Objective

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} |\delta(i,j)^2 - \|y_i - y_j\|^2|$$

$$\text{Admissibility} \rightarrow \delta(i,j)^2 \geq \|y_i - y_j\|^2$$

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} (\delta(i,j)^2 - \|y_i - y_j\|^2)$$

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} \delta(i,j)^2 - \sum_{i,j} W_{ij} \|y_i - y_j\|^2$$

Defining loss

Simplifying the Objective

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} \left| \delta(i,j)^2 - \|y_i - y_j\|^2 \right|$$

$$\text{Admissibility} \rightarrow \delta(i,j)^2 \geq \|y_i - y_j\|^2$$

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} \left(\delta(i,j)^2 - \|y_i - y_j\|^2 \right)$$

$$\mathcal{L}(Y) = \sum_{i,j} W_{ij} \delta(i,j)^2 - \sum_{i,j} W_{ij} \|y_i - y_j\|^2$$

$$\boxed{\underset{Y}{\textbf{maximize}} \sum_{i,j} W_{ij} \|y_i - y_j\|^2}$$

Connections

Nonlinear dimensionality reduction

The full optimization problem:

$$\begin{aligned} \textbf{maximize}_{Y} \quad & \sum_{i,j} W_{ij} \|y_i - y_j\|^2 \\ \textbf{subject to} \quad & \forall (i,j) \in E \ \|y_i - y_j\| \leq d(i,j) \end{aligned}$$

CONNECTIONS

NONLINEAR DIMENSIONALITY REDUCTION

The full optimization problem:

$$\begin{aligned} \underset{Y}{\textbf{maximize}} \quad & \sum_{i,j} W_{ij} \|y_i - y_j\|^2 \\ \textbf{subject to} \quad & \forall (i,j) \in E \ \|y_i - y_j\| \leq d(i,j) \end{aligned}$$

This is a weighted generalization of Weinberger *et al.*'s Maximum Variance Unfolding (MVU)

Connections

Nonlinear dimensionality reduction

The full optimization problem:

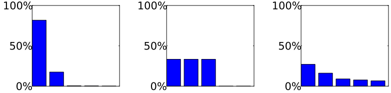
$$\begin{aligned} \mathop{\textbf{maximize}}_{Y} \quad & \sum_{i,j} W_{ij} \|y_i - y_j\|^2 \\ \textbf{subject to} \quad & \forall (i,j) \in E \ \|y_i - y_j\| \leq d(i,j) \end{aligned}$$

This is a weighted generalization of Weinberger *et al.*'s Maximum Variance Unfolding (MVU)

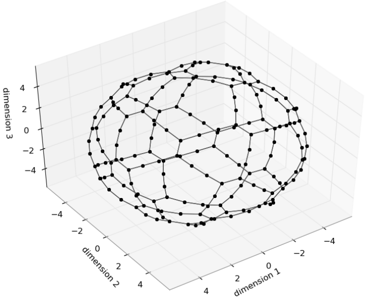
Heuristic learning is linked to manifold learning

Manifold learning

- What "dimensionality" will hold a search graph?



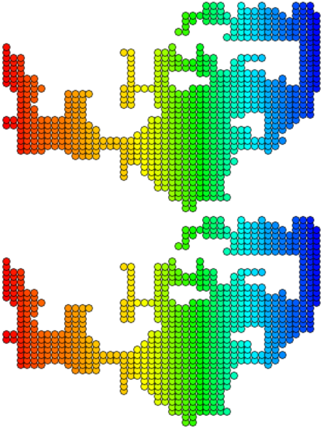
- Visualization may shed new light on problems



Differential heuristics

Ng & Zhang '02, Goldberg & Harrelson '05, Sturtevant *et al.*'09

Imagine "hooking" the graph on a pivot state



Connections

Differential Heuristics

Our approach recovers differential heuristics when $W = W_{\text{diff}}$:

Connections

Differential Heuristics

Our approach recovers differential heuristics when $W = W_{\text{diff}}$:

Let "pivot" be state 1 and $n = 4$:

$$W_{\text{diff}} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & -1/3 & -1/3 \\ 1 & -1/3 & 0 & -1/3 \\ 1 & -1/3 & -1/3 & 0 \end{bmatrix}$$

CONNECTIONS

DIFFERENTIAL HEURISTICS

Our approach recovers differential heuristics when $W = W_{\text{diff}}$:

Let "pivot" be state 1 and $n = 4$:

$$W_{\text{diff}} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & -1/3 & -1/3 \\ 1 & -1/3 & 0 & -1/3 \\ 1 & -1/3 & -1/3 & 0 \end{bmatrix}$$

- Push points *away* from the pivot state (weight 1)

Connections

Differential Heuristics

Our approach recovers differential heuristics when $W = W_{\text{diff}}$:

Let "pivot" be state 1 and $n = 4$:

$$W_{\text{diff}} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & -1/3 & -1/3 \\ 1 & -1/3 & 0 & -1/3 \\ 1 & -1/3 & -1/3 & 0 \end{bmatrix}$$

- Push points *away* from the pivot state (weight 1)
- and pull points *into* each other (weight 1/(n-1))

CONNECTIONS

DIFFERENTIAL HEURISTICS

Our approach recovers differential heuristics when $W = W_{\text{diff}}$:

Let "pivot" be state 1 and $n = 4$:

$$W_{\text{diff}} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & -1/3 & -1/3 \\ 1 & -1/3 & 0 & -1/3 \\ 1 & -1/3 & -1/3 & 0 \end{bmatrix}$$

- Push points *away* from the pivot state (weight 1)
- and pull points *into* each other (weight 1/(n-1))

Can this be improved?

CONNECTIONS

DIFFERENTIAL HEURISTICS

Our approach recovers differential heuristics when $W = W_{\text{diff}}$:

Let "pivot" be state 1 and $n = 4$:

$$W_{\text{diff}} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & -1/3 & -1/3 \\ 1 & -1/3 & 0 & -1/3 \\ 1 & -1/3 & -1/3 & 0 \end{bmatrix}$$

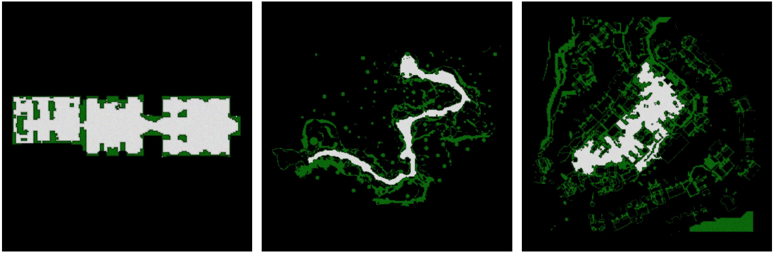
- Push points *away* from the pivot state (weight 1)
- and pull points *into* each other (weight 1/(n-1))

Can this be improved?

$$W^{+}_{\text{diff}} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & \epsilon & \epsilon \\ 1 & \epsilon & 0 & \epsilon \\ 1 & \epsilon & \epsilon & 0 \end{bmatrix}$$

Let's experiment with $\epsilon = 10^{-3}$...

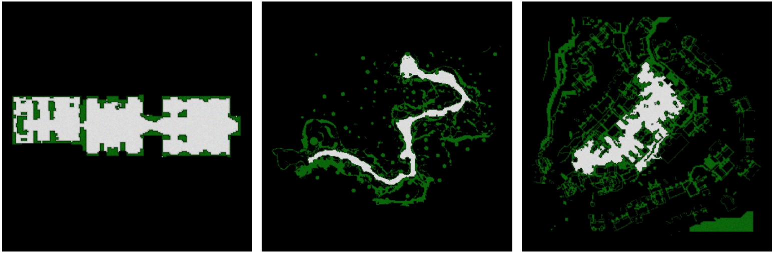
Pathplanning experiment



Dragon Age: Origins (BioWare)
www.movingai.com

- Maps with 168–6,240 states: standard problem sets

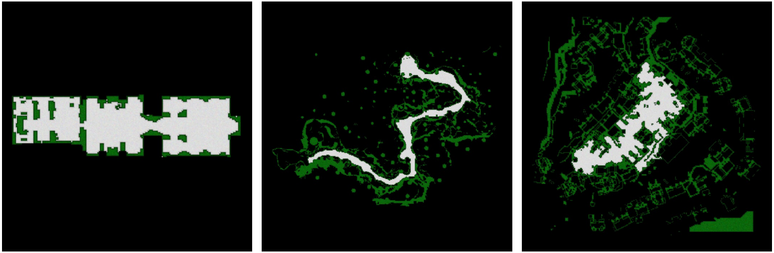
PATHPLANNING EXPERIMENT



Dragon Age: Origins (BioWare)
www.movingai.com

- Maps with 168–6,240 states: standard problem sets
- Count the nodes A* [Hart *et al.* 68] expands to find a path

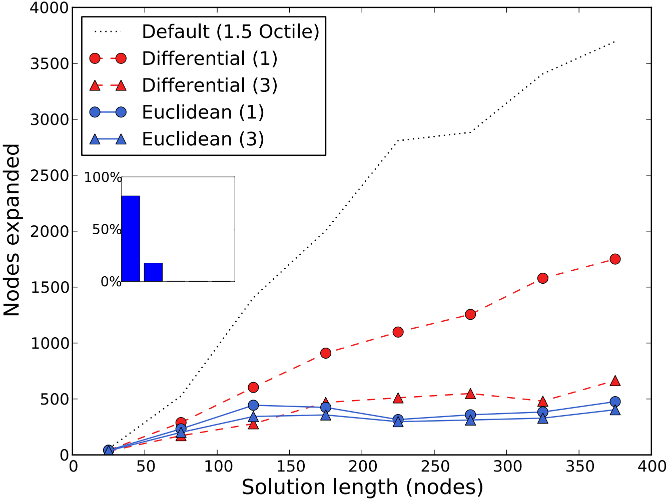
PATHPLANNING EXPERIMENT



Dragon Age: Origins (BioWare)
www.movingai.com

- Maps with 168–6,240 states: standard problem sets
- Count the nodes A* [Hart *et al.* 68] expands to find a path
- Compare W_{diff} and W^{+}_{diff}:

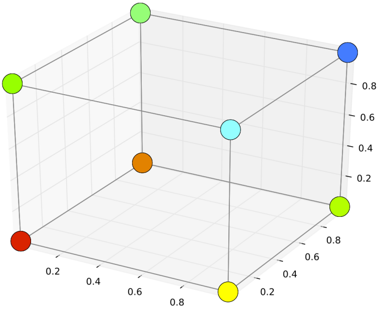
Pathplanning Results



Sets of 1 and 3 differential heuristics *vs.*
Sets of 1 and 3 Euclidean heuristics ($W = W_{\text{diff}}$)

CUBE WORLD EXPERIMENT

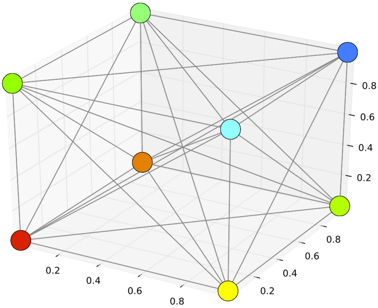
Differential heuristics' weakness: high dimensionality



- Octile grid world, generalized to higher dimensions

CUBE WORLD EXPERIMENT

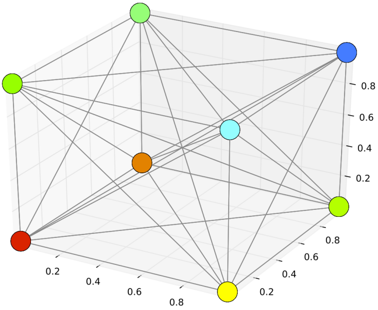
Differential heuristics' weakness: high dimensionality



- Octile grid world, generalized to higher dimensions
  - Agent can increment any/all coordinates by 1 each turn

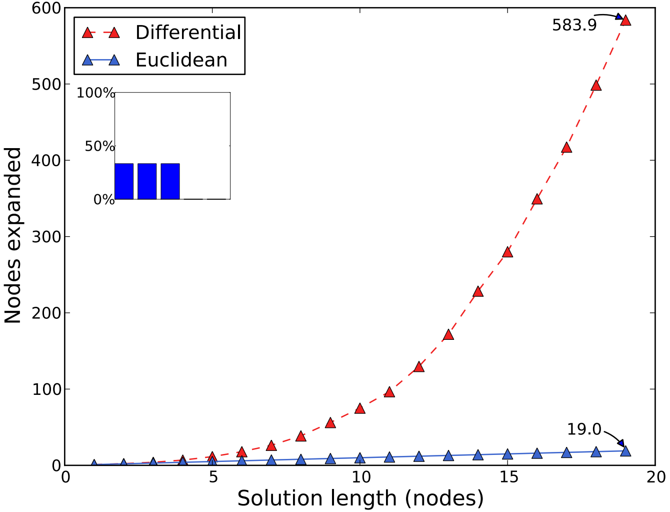
Cube world experiment

Differential heuristics' weakness: high dimensionality



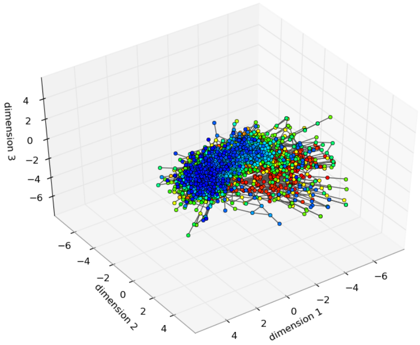
- Octile grid world, generalized to higher dimensions
  - Agent can increment any/all coordinates by 1 each turn
- Transition costs are edge lengths

Cube World Results ($20 \times 20 \times 20$)



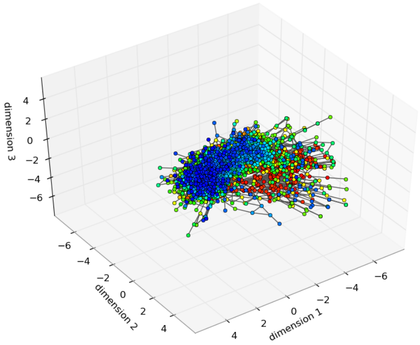
Set of 4 differential heuristics *vs.*
one 3-dimensional Euclidean heuristic ($W_{ij} = 1$)

WORD GRAPH EXPERIMENT



- 4,820 states representing four-letter words

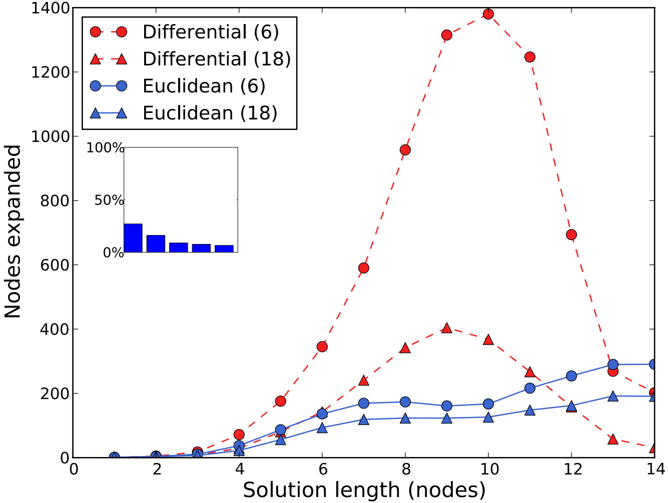
WORD GRAPH EXPERIMENT



- 4,820 states representing four-letter words
- Find shortest sequence of 1-letter changes turning start word into goal word

$$\text{fore} \to \text{fork} \to \,? \to \text{back}$$

Word Graph Results



Sets of 6 and 18 differential heuristics *vs.*
6 and 18-dimensional Euclidean heuristics ($W_{ij} = 1$)

SUMMARY

Euclidean Heuristic Optimization

A novel way to build admissible/consistent heuristics

- principled link to manifold learning
- generalization of differential heuristics
- promising empirical results on small problems

(Thank you: Ariel Felner, our anonymous reviewers, NSERC and iCore)