Local Routing in a new Indefinitely Scalable Asynchronous Architecture

Trent Small
School of Computer Science
University of New Mexico
Albuquerque, New Mexico 87131–0001
Email: tsmall1@unm.edu

***Abstract*—Local routing is a problem which most of us face on a daily basis. This study proposes a routing method based on road signs in a procedurally generated city and shows its overall efficiency in a variety of environments. We show that this method is efficient in only some city environments.**

I. INTRODUCTION

II. MODEL

This simulation models a city with robust repairing properties, utilising techniques common to self-organization [1]. This increases the robustness of the city and allows traffic to continue in an organized manner.

A. City Growth

The city first begins with a single ELEMENT_CITY_STREET, which simply tries to go in a given cardinal direction. If the street encounters an Element other than something which belongs in the street (namely an ELEMENT_CITY_CAR, an ELEMENT_CITY_INTERSECTION, or another Street), it replaces that Element with either a Street or an Intersection depending on its Intersection odds parameter. The roads continue to grow indefinitely.

As the streets grow, they place ELEMENT_CITY_SIDEWALK elements in the sites perpendicular to the direction they are travelling. Once a Sidewalk has been created, it waits for a parameterized number of events before creating an ELEMENT_CITY_BUILDING on the side opposite the Street. The sidewalks are produced solely by the Streets and do not reproduce.

Once an ELEMENT_CITY_BUILDING has been created, it begins to reproduce along the Sidewalk. Each building has a TYPE parameter, symbolizing the type of building it is. As a practical example, each grocery store, be it a Kroger or an Albertsons, is still a grocery store. These buildings only grow up to a maximum size, insisting diversity on every city block. Once a building has been created, it also gains the capacity to create an ELEMENT_CITY_CAR in place of a nearby Street.

The ELEMENT_CITY_CAR simply drives along a Street by swapping in the direction of the Street it was created in place of. A car has a limited amount of events before it "runs out of gas". If a Car has an event next to a Building of the type that it is trying to go to, it is consumed and the remaining fuel is reported.

B. Routing

This system grows a city with Streets, Cars, and Buildings, but so far no routing has been described. Routing is the job of both Intersections and Sidewalks.

Sidewalks build a local map of the city near them. This is stored as an array of 2-bit numbers of length n, where n is the number of building types in the city. This array stores the distance of this Sidewalk from the building of each type, measured in city blocks. This is populated as such:

```
if I border a building then
  | Set my map[building type] to 0
end
for each building type t do
  | Get the minimum m of my neighbor's maps for t
  | Set my map[t] to MIN(map[t], m).
  | if There is a Sidewalk across the street from me, s
  | then
  |   | Set my map[t] to MIN(map[t], (s[t]+1)).
  | end
end
```

Algorithm 1: Sidewalk Mapping

This map is ultimately read by an Intersection when a Car is waiting. The Intersection examines the Sidewalks surrounding neighboring Streets and places the Car on the Street which is between the blocks closest to its destination. This routing model points each Car in the right direction.

III. CONCLUSION

The conclusion goes here.

ACKNOWLEDGMENT

The authors would like to thank...

REFERENCES

[1] Misteli, T. *The Concept of Self-organization in Cellular Architecture. The Journal of Cell Biology* 155.2 (2001): 181–86. Web.