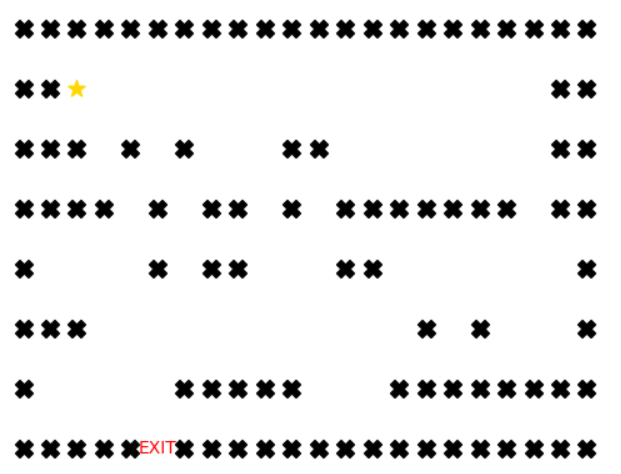
**So sánh 2 thuật toán Greedy Best First Search với A\*:**

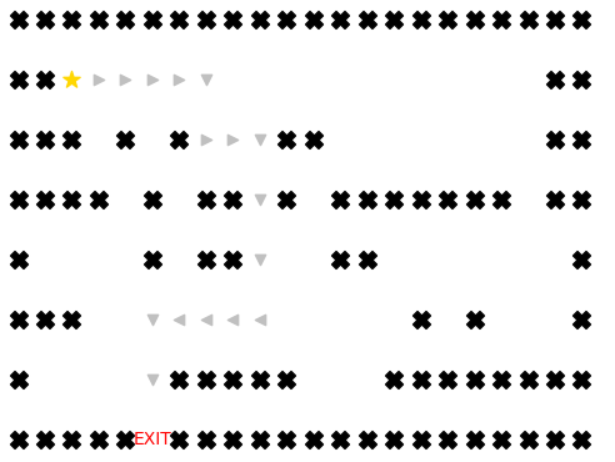
Greedy Best First Search tìm kiếm theo hàm heuristic (Khoảng cách từ node này đến đích). Ở mỗi lần di chuyển ta sẽ thêm vào priority queue các node con liền kề mà có trạng thái chưa được đi qua. Priority queue được xếp theo thứ tự ưu tiên cao nhất là những node có heuristic nhỏ nhất( tức là những node chúng ta ước tính gần đích nhất). Rồi cứ tiếp tục lấy ra các node rồi lại thêm node con liền kề như vậy cho đến khi chúng ta lấy được node đích( Exit).

Còn A\* chỉ khác GBFS ở chỗ không chỉ lấy hàm heuristic mà chúng ta còn cộng cả chi phí từ điểm xuất phát đến node đang xét.

**Map 1:**



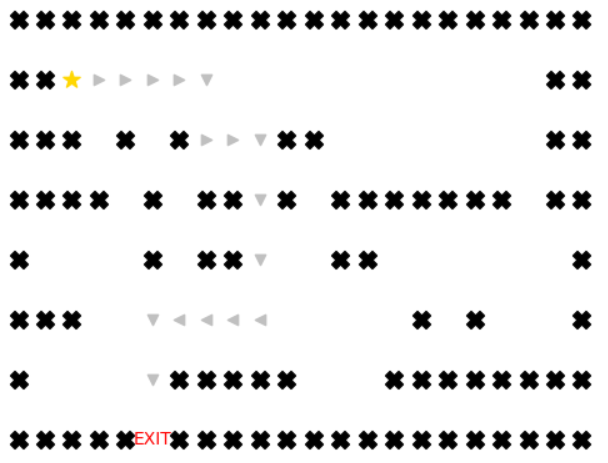
Khi thực hiện thuật toán GBFS, ta được đường đi:



Chi phí đường đi và tổng số trạng thái phải xét của thuật toán GBFS:



Khi thực hiện thuật toán A\*, ta được đường đi:



Chi phí đường đi và tổng số trạng thái phải xét của thuật toán A\*:



Nhận xét: Ở bản đồ này ta thấy chi phí đường đi giống nhau và số trạng thái phải xét cũng xấp xỉ nhau nên 2 thuật toán ở trường hợp này không có nhiều sự khác biệt.

**Map 2:**

Ảnh có chứa bàn

Mô tả được tạo tự động

Khi ta chạy thuật toán Greedy Best First Search, ta được đường đi như sau:

Ảnh có chứa bàn

Mô tả được tạo tự động

Chi phí đường đi và tổng số trạng thái phải xét:



Khi chạy A\*, ta được đường đi:

Ảnh có chứa bàn

Mô tả được tạo tự động

Chi phí đường đi và tổng số trạng thái phải xét:



Nhận xét: Nhìn chung ở map này cả 2 thuật toán đều tìm được đường đi ngắn nhất giống nhau. Còn thời gian thực hiện thì A\* có vẻ nhiều hơn 1 chút nhưng không đáng kể.

**Map 3:**

Ảnh có chứa bàn

Mô tả được tạo tự động

Khi thực hiện GBFS, ta được đường đi:

Ảnh có chứa bàn

Mô tả được tạo tự động

Chi phí đường đi và tổng số trạng thái phải xét:



Khi thực hiện A\*, ta được đường đi:

Ảnh có chứa bàn

Mô tả được tạo tự động

Chi phí đường đi và tổng số trạng thái phải xét:



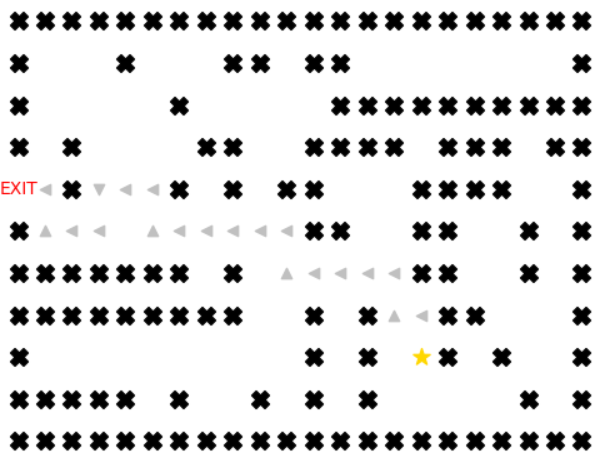
Ở bản đồ này ta thấy thuật toán A\* tối ưu hơn 1 cách rõ rệt khi chi phí đường đi giảm gần 1 nửa so với GBFS và chi phí thời gian cũng nhỏ hơn.

**Map 4:**

Ảnh có chứa bàn

Mô tả được tạo tự động

Khi chạy GBFS, ta được đường đi:



Chi phí và tổng số trạng thái phải xét:



Khi chạy A\*, ta được đường đi:

Ảnh có chứa bàn

Mô tả được tạo tự động

Chi phí và tổng số trạng thái phải xét:



Nhận xét: Đối với bản đồ này, thuật toán A\* vẫn tìm được đường đi tối ưu hơn so với GBFS.

**Map 5 (Map lớn):**

Ảnh có chứa bàn

Mô tả được tạo tự động

Khi thực hiện GBFS, ta được đường đi:

Ảnh có chứa văn bản, người, ảnh chụp màn hình

Mô tả được tạo tự động

Chi phí đường đi và tổng trạng thái phải xét của GBFS là:



Khi thực hiện A\*, ta được đường đi:

Ảnh có chứa văn bản, người, ảnh chụp màn hình

Mô tả được tạo tự động

Chi phí đường đi và tổng số trạng thái phải xét là:



Nhận xét: Ở bản đồ này ta thấy thuật toán A\* vẫn tìm được đường đi ngắn hơn nhưng mà số các trạng thái phải xét lại nhiều hơn đáng kể (gấp đôi so với GBFS).

**Nhận xét chung:**

Ta thấy thuật toán A\* là 1 thuật toán tối ưu về tìm kiếm đường đi còn thuật toán GBFS lại quá phụ thuộc vào hàm heuristics. Mà khi hàm heuristics không tốt (ước tính không chuẩn) sẽ dẫn đến những đường đi không tối ưu.