1. Giới thiệu
   1. Mở đầu
   2. Yêu cầu & mục tiêu đề tài
      1. Yêu cầu
      2. Mục tiêu

Hệ thống gồm 3 phần

* Extract
* Match
* Query (Search)

1. Phân tích
2. Dữ liệu cần extract:

Dựa trên yêu cầu của hệ thống: thu thập dữ liệu liên quan đến địa chỉ. Nhóm đã xác định các thuộc tính liên quan đến địa chỉ cần extract

* Tên cơ sở tại địa chỉ đã cho.
* Địa chỉ chính xác của cơ sở (VD: phòng, tầng …)
* E-mail (nếu có)
* Website (nếu có)
* Số điện thoại (nếu có)
* Số fax (nếu có)
* Ngành nghề kinh doanh (nếu có)

1. Extract

* Nguồn dữ liệu của hệ thống : web pages từ các website khác nhau.
* Thuận lợi: nguồn dữ liệu có cấu trúc.
* Page có 2 dạng:

+ List page: chứa 1 hoặc nhiều list, mỗi list gồm nhiều data record được định dạng theo cấu trúc giống nhau.

+ Detail page: page chứa thông tin chi tiết về 1 đối tượng

* mục tiêu là tự động extract dữ liệu từ cả 2 dạng page này.
* Các giải pháp hiện tại:

+ Manual approach

+ Wrapper induction

+ Automatic extraction

* + 1. Manual approach

Người lập trình quan sát webpage và source code của nó để tìm ra pattern, sau đó viết chương trình để lấy dữ liệu, có thể sử dụng các công cụ : xpath… để việc lấy dữ liệu dễ dàng hơn. Chỉ ứng với từng website cụ thể, không thể mở rộng.

* + 1. Wrapper induction

Dựa vào một số page được đánh dấu sẵn vị trí của data record, hệ thống tự động rút ra quy luật và dùng luật này để extract data từ những page có cấu trúc tương tự.

Nhược điểm:

- Tốn công sức và thời gian cho việc đánh dấu vị trí data record bằng tay.

- Chi phí cao khi áp dụng cho nhiều site khác nhau.

* + 1. Automatic extraction

Cho trước một hoặc một số page, hệ thống sẽ tự động tìm ra pattern để extract data. Cách tiếp cận này loại bỏ việc đánh dấu vị trí bằng tay, do đó có thể áp đụng cho 1 số lượng lớn page và site.

Một giải thuật thuộc cách tiếp cận này là giải thuật NET của tác giả Biu Ling. Giải thuật này dựa trên sự so trùng DOM tree để tìm ra pattern cho việc extract data. Giải thuật áp dụng cho list page, tuy nhiên có thể dễ dàng điều chỉnh để áp dụng cho detail page bằng cách tạo một DOM tree từ các detail page này.

Nhược điểm dễ nhận thấy của giải thuật là chưa thể tạo được attribute name cho dữ liệu được extract => cần có cơ chế nhận dạng, gán nhãn cho dữ liệu. Nhóm sử dụng heuristic dựa trên các đặc điểm của dữ liệu để làm việc này.

1. Match
   * 1. Cơ sở dữ liệu hiện có

Cơ sở dữ liệu hiện có là một DB chứa đựng thông tin về các khu đất được đánh địa chỉ tại TpHCM kèm theo tọa độ cụ thể của khu đất.

Chi tiết về cơ sở dữ liệu này được biểu diễn trong ERD dưới đây

ERD trên đã được map thành các bảng như sau:

* + 1. Cách thức map:

Việc map các đối tượng dữ liệu đã được extract vào cơ sở dữ liệu hiện có dựa trên thuộc tính về địa chỉ chính xác của đối tượng dữ liệu.

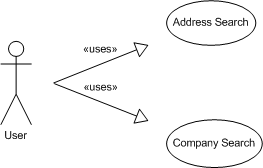
Quá trình map được thực hiện bằng 2 bước:

- Phân tích chuỗi địa chỉ chính xác để có được số nhà, tên đường, phường, quận.

- Map đối tượng dữ liệu vào 1 địa chỉ trong cơ sở dữ liệu hiện có dựa trên 4 thuộc tính vừa được phân tích.

1. Query

Dựa trên yêu cầu xây dựng 1 website cho phép người dùng tìm kiếm trên dữ liệu địa chỉ vừa extract. Nhóm dự định xây dựng website có các chức năng như sơ đồ use-case dưới đây



Trong đó:

* Chức năng 1: Address Search dùng để tìm địa chỉ cụ thể
* Chức năng 2: Company Search dùng để tìm công ty , dịch vụ hay một cửa hàng nào đó

1. Giải thuật NET:
2. Tree Matching và giải thuật Simple Tree Matching
3. Multiple Alignment và giải thuật Partial Tree Alignment
   1. Multiple Alignment

Để tìm ra mô hình (pattern) lặp từ chuỗi HTML dựa vào khoảng cách chuỗi chỉnh sửa (string edit distance) hoặc kết hợp cây (tree matching) , chúng ta cần sự sắp xếp của các chuỗi và cây đó. Một trang web thường chứa hơn hai data record, do đó có nhiều hơn hai chuỗi hoặc cây cần phải được liên kết. Việc tạo ra một sự liên kết tổng thể của tất cả các chuỗi hoặc cây là rất quan trọng. Đó chính là Multiple Alignment .

Trong giới hạn báo cáo này , nhóm sẽ đề cập đến giải thuật Partial Tree Alignment , một phương pháp sắp xếp theo kinh nghiệm .

* 1. Partial Tree Alignment
     1. Partial Alignment of Two Trees

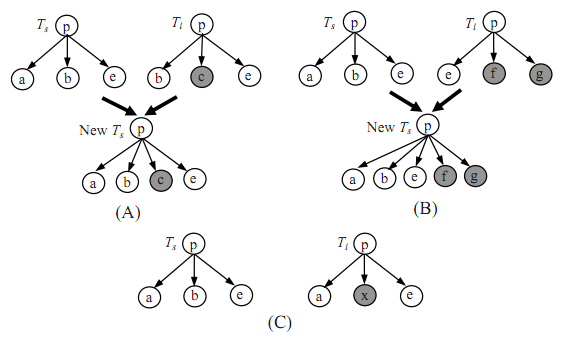
Trước khi đến với một giải thuật đầy đủ , ta sẽ nhìn vào sự liên kết một phần của 2 cây . Sau khi 2 cây Ts và Ti được kết , một số nút trong Ti có thể được liên kết với các nút tương ứng của Ts vì chúng phù hợp với nhau . Đối với những nút trong Ti không phù hợp, ta sẽ chèn chúng vào Ts với các dữ liệu là tùy chọn.Có hai tình huống có thể khi chèn một nút mới vi từ Ti vào Ts, tùy thuộc vào vị trí trong Ts có thể được xác định duy nhất để chèn vi.Thay vì xem xét một nút duy nhất vi, chúng ta có thể xem xét một tập các nút liền kề Vj ... Vm chưa được liên kết từ Ti với nhau. Không mất tính tổng quát, ta giả định rằng nút cha của Vj ... Vm có sự trùng khớp trong Ts và ta muốn chèn Vj ... Vm vào Ts dưới cùng một nút cha. Ta chỉ chèn Vj ... Vm vào Ts nếu một vị trí để chèn Vj ... Vm có thể được xác định duy nhất trong Ts.Nếu không, chúng sẽ không được chèn vào Ts (unaligned). Các vị trí để chèn Vj ... Vm có thể chỉ được quyết định bởi :

1. Nếu Vj ... Vm có 2 nút kết cận trong Ti , một bên phải và một bên trái ,thì chúng trùng với 2 nút liên tiếp trong Ts ( Hình A)

2. Nếu Vj ... Vm chỉ có 1 nút trái x thuộc Ti và x lại trùng với nút phải nhất trong Ts , thì Vj ... Vm có thể được thêm vào sau nút x trong Ts ( Hình B )

3. Nếu Vj ... Vm chỉ có 1 nút phả x thuộc Ti và x lại trùng với nút trái nhất trong Ts , thì Vj ... Vm có thể được thêm vào trước nút x trong Ts

Các trường hợp còn lại , ta không thể xác định được vị trí duy nhất để thêm vào Ts ( Hình C )



* + 1. Partial Alignment of Multiple Trees

**Algorithm** PartialTreeAlignment(S)

1. Sort trees in S in descending order of the number of unaligned data items;

2. Ts  the first tree (which is the largest) and delete it from S;

3. R 

4. **while** (S ≠ ) **do**

5. Ti  select and delete next tree from S; // follow the sorted order

6. STM(Ts, Ti); // tree matching

7. AlignTrees(Ts, Ti); // based on the result from line 6

8. **if** Ti is not completely aligned with Ts **then**

9. **if** InsertIntoSeed(Ts, Ti) **then** // True: some insertions are done

10. S S  R;

11. R 

12 **endif;**

13. **if** there are still unaligned items in Ti that are not inserted into Ts **then**

14. R  R  {Ti}

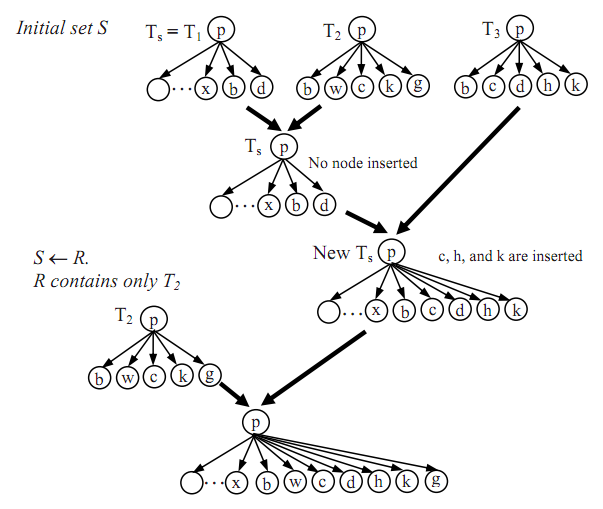
15. **endif;**

16. **endif;**

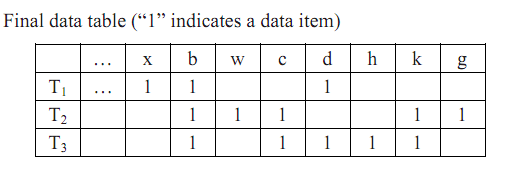
17. **endwhile;**

18. Output data fields from each Ti to a data table based on the alignment results.

Đây là giải thuật đầy đủ đối với trường hợp nhiều cây dựa trên giải thuật partial alignment trên 2 cây . Dòng 1-2 ( Trong giải thuật) sẽ tìm cây chứa nhiều data items nhất . Nó được dùng như cây giống Ts ( Seed Tree ) . Dòng 3 khởi tạo R , cái sẽ chứa những cây không hoàn toàn được sắp thẳng với Ts trong mỗi vòng lặp . Dòng 4 bắt đầu vòng lặp While để sắp xếp mỗi các cây với Ts . Dòng 5 lấy chưa được sắp xếp kế tiếp , dòng 6 thực hiện việc kết nối . Dòng 7 tìm tất cả các cặp trùng khớp bằng cách lần theo ma trận kết quả từ dòng 6 . Hàm này tương tự như việc sắp xếp 2 chuỗi sử dụng khoảng cách chỉnh sửa . Dòng 8 và dòng 9 cố găng thêm những nút không trùng vào Ts . Đây là giải thuật partial tree alignment đã nêu ở trên . Dòng 13 – 14 thêm T2 vào R , là danh sách các cây cần được kết hợp lại vì một số data items chưa được sắp xếp và chưa được thêm vào Ts . Dòng 10 – 11 đưa những cây trong R vào S và khởi tạo lại R . Dòng 18 xuất ra các data items từ mỗi cây theo sự sắp xếp . Hình bên dưới minh hoạ cho giải thuật trên







Trong thực tế , để có một giải thuật hoàn thiện , một lời gọi đệ quy nên được thêm vào sau dòng 17 để sửa lý trường hơp R ≠ 0 . Ba dòng sau có thể được dùng

18. if R  then

19. PartialTreeAlignment(R)

20. endif

Điều này sẽ đảm bảo được trường hợp có những item chưa được sắp xếp và thêm vào . Ta cần lưu ý 2 điểm đối với giải thuật hoàn thiện này . Thứ 1 , lời gọi đệ quy sẽ dừng ngay cả khi không có một sắp xếp nào hay một việc thêm nào được tạo ra đối với cây giống bởi vì cây giống đã bị xoá trong mỗi lần gọi đệ quy và do đó R sẽ càng lúc càng nhỏ . Thứ 2 , giải thuật có thể tìm thấy nhiều mẫu trong dữ liệu . Cây giống từ mỗi lần gọi đệ quy sẽ đại diện cho một mẫu khác nhau .

1. Giải thuật NET
2. Thiết kế và hiện thực:
   1. Tổng quan hệ thống

Hệ thống gồm 2 hệ thống con Extractor và Query như hình vẽ sau

Database

Match

Query

(Website)

Extractor

Extractor rút trích thông tin địa chỉ từ nguồn dữ liệu về database.

Website sử dụng database để cung cấp thông tin cho người sử dụng.

* 1. Công cụ sử dụng

Python

ASP.NET

Microsoft SQL Server

* 1. Extractor & Match
     1. Thiết kế kiến trúc

Sub-system extractor gồm các module sau

- Matching: hiện thực giải thuật Simple Tree Matching

- Alignment: hiện thực giải thuật Partial Tree Alignment

- Wildcard: chứa các hàm support cho việc hiện thực wildcard trong pattern

- Label: các hàm dùng gán nhãn dữ liệu

- Support: các hàm phụ trợ khác

* + 1. Thiết kế database

ERD

Bảng

* + 1. Hiện thực
  1. Query
     1. Cơ sở dữ liệu

Phần website sử dụng cơ sở dữ liệu phần thiết kế Extractor đã tạo, để tiện trong việc truy vấn, nhóm cần bổ sung thêm các view .

Từ quan hệ của các bảng ta thấy một công ty có thể có nhiều số điện thoại , nhiều số fax và nhiều ngành nghề khác nhau nên để dễ dàng cho việc tạo view chính ta cần tạo trước các view nhỏ sau

- View : CongTy\_DienThoai\_VW

CREATE VIEW CongTy\_DienThoai\_VW

AS

SELECT i.CongTyId, STUFF(g.y, 1, 1, '') AS SoDienThoai

FROM

(SELECT CongTyId FROM tbl\_CongTy\_DienThoai GROUP BY CongTyId ) AS i

CROSS APPLY (

SELECT DISTINCT ',' + CAST(SoDienThoai AS NNVARCHAR(11))

FROM tbl\_CongTy\_DienThoai AS s

WHERE s.CongTyId = i.CongTyId

ORDER BY ',' + CAST(SoDienThoai AS NVARCHAR(11))

FOR XML PATH('')

) AS g(y);

View tạo được có dạng

|  |  |
| --- | --- |
| CongTyId | SoDienThoai |
| 1 | 0838225223,0839326280 |
| 2 | 0838272811,0838272812,0838276027 |

- View 2 : CongTy\_Fax\_VW

CREATE VIEW CongTy\_Fax\_VW

AS

SELECT i.CongTyId, STUFF(g.y, 1, 1, '') AS SoFax

FROM

(SELECT CongTyId FROM tbl\_CongTy\_Fax GROUP BY CongTyId ) AS i

CROSS APPLY (

SELECT DISTINCT ',' + CAST(SoFax AS NVARCHAR(11))

FROM tbl\_CongTy\_Fax AS s

WHERE s.CongTyId = i.CongTyId

ORDER BY ',' + CAST(SoFax AS NVARCHAR(11))

FOR XML PATH('')

) AS g(y)

View tạo được có dạng :

|  |  |
| --- | --- |
| CongTyId | Sofax |
| 1 | 0822172675,0838127196 |
| 2 | 0838218590, 0838298762, 0839600015 |

- View 3 : CongTy\_NganhNghe\_VW

CREATE VIEW CongTy\_NganhNghe\_VW

AS

SELECT i.CongTyId, STUFF(g.y, 1, 1, '') AS NganhNghe

FROM

(SELECT m.CongTyId FROM (select CongTyId,Ten

FROM tbl\_CongTy\_NganhNghe ctnn

LEFT JOIN tbl\_NganhNghe nn

ON ctnn.NganhNgheId = nn.Id) as m GROUP BY CongTyId ) AS i

CROSS APPLY (

SELECT DISTINCT ',' + CAST(Ten AS NVARCHAR(11))

FROM (select CongTyId,Ten

FROM tbl\_CongTy\_NganhNghe ctnn

LEFT JOIN tbl\_NganhNghe nn

ON ctnn.NganhNgheId = nn.Id) AS s

WHERE s.CongTyId = i.CongTyId

ORDER BY ',' + CAST(Ten AS NVARCHAR(11))

FOR XML PATH('')

) AS g(y)

View tạo được có dạng :

|  |  |
| --- | --- |
| CongTyId | NganhNghe |
| 1 | Giày Dép,Mua Sắm |
| 2 | Dịch Vụ,Ngành Nước |

-View 4 : SEARCH\_DIACHI\_VW

CREATE VIEW SEARCH\_DIACHI\_VW

AS

SELECT ct.Id,Ten,DiaChiChinhXac,SoDienThoai,Email,SoFax,Website,Rank,SoNha,Duong,Phuong,Quan,DuongKhongDau,QuanKhongDau,NganhNghe,X,Y

FROM tbl\_CongTy ct

RIGHT JOIN tbl\_DiaChi dc

ON ct.DiaChiId = dc.Id

LEFT JOIN CongTy\_DienThoai\_VW dt

ON ct.Id = dt.CongTyId

LEFT JOIN CongTy\_Fax\_VW sf

ON ct.Id = sf.CongTyId

LEFT JOIN CongTy\_NganhNghe\_VW nn

ON ct.Id = nn.CongTyId

View này là tổng hợp từ các view trên và phục vụ cho chức năng search .

* + 1. Hiện thực

Website Location Finder được xây dựng với 2 chức năng chính : Tìm địa chỉ và tìm công ty

- Chức năng 1 : Tìm địa chỉ

Để thực hiện chức năng tìm địa chỉ ta cần nhập vào textbox search theo cú pháp : Số Nhà , Tên đường

- Chức năng 2 : Tìm công ty

Để thực hiện chức năng tìm công ty ta chỉ cần nhập vào textbox search tên công ty cần tìm

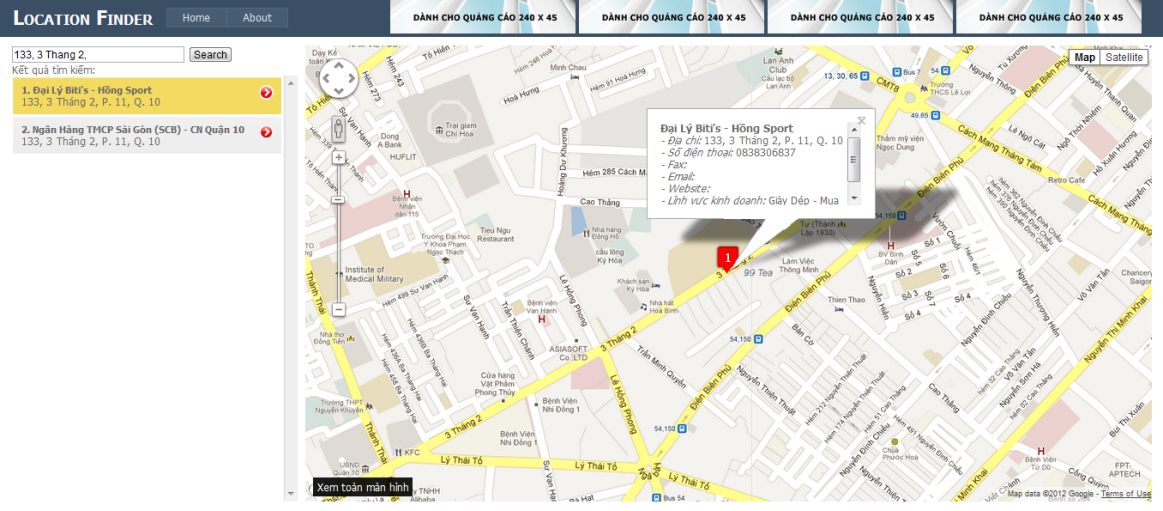
Khi chuỗi query địa chỉ ( Ví dụ : 133, 3 Thang 2 ) hay tên công ty (Ví dụ : Cửa hàng ) được nhập vào hệ thống sẽ tìm kiếm trong view SEARCH\_DIACHI\_VW với điều kiện :

WHERE DuongKhongDau LIKE N'%133 3 Thang 2%' OR (SoNha LIKE '%133%' AND DuongKhongDau LIKE N'%3 Thang 2%') OR Ten LIKE N'%133 3 Thang 2%'

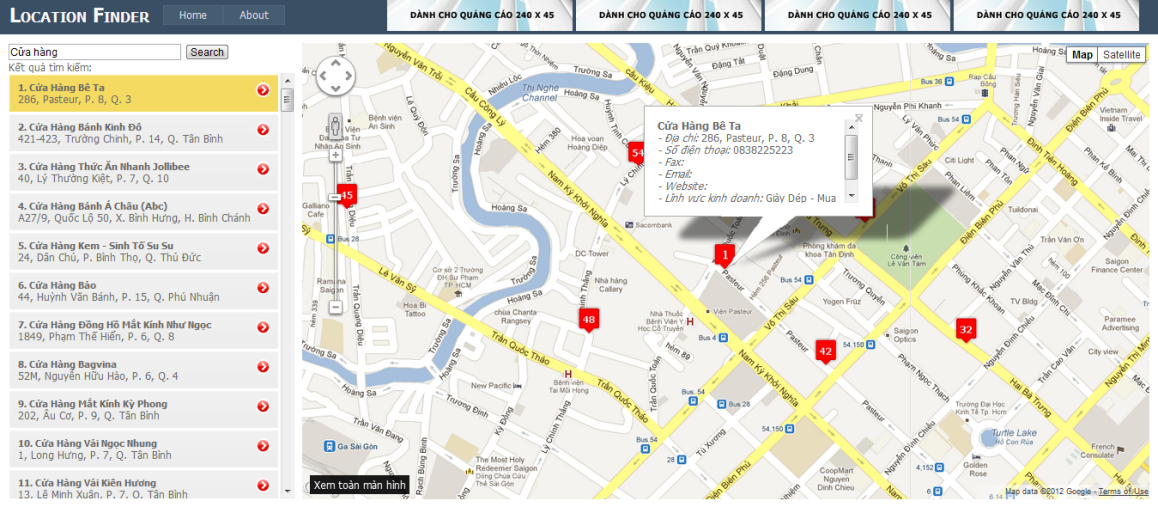
Đồng thời tăng giá trị cột Rank tương ứng với địa chỉ hay công ty đó trong CSDL, giúp tăng tốc việc Autocomplete từ client :

UPDATE tbl\_DiaChi SET RankDC +=1 WHERE DuongKhongDau LIKE N'%133 3 Thang 2%' OR (SoNha LIKE '%133%' AND DuongKhongDau LIKE N'%3 Thang 2%')

1. Sử dụng:
2. Extractor
3. Website :
4. Tìm địa chỉ



1. Tìm công ty



Sau khi nhập vào textbox địa chỉ cần tìm và click button search , một danh sách công ty được tìm thấy sẽ hiện ra đồng thời vị trí cũng được xác định trên bản đồ . Bên cạnh đó , khi click lên mỗi kết quả trong danh sách thì thông tin cụ thể của công ty đó sẽ được hiện rõ trên bản đồ .

Thông tin chi tiết bao gồm :

- Tên công ty : *Beauty Salon*

- Địa chỉ : *203, Nguyễn Thượng Hiền, P. 4, Q. 3*

- Số điện thoại : *0839747765*

- Số fax : *0838112446*

- Email : *info@sieuthitocdep.com*

- Website : *http://www.sieuthitocdep.com*

- Lĩnh vực kinh doanh : *Cắt - Uốn Tóc, Làm Đẹp*

1. Đánh giá:
2. Các kết quả đạt được
3. Các hạn chế của hệ thống
4. Hướng phát triển