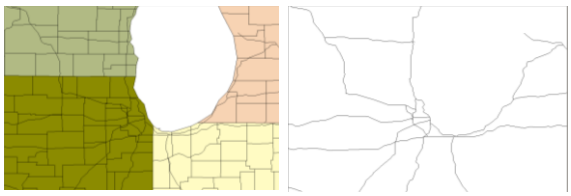
Spatial Networks

Spatial Networks and Graphs

Networks



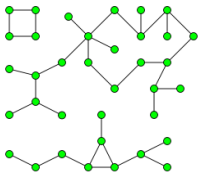
What other networks are there?

What do we need to model this?

Graphs

G= (V,E)

- V: vertices (nodes)
- E: edges (links)



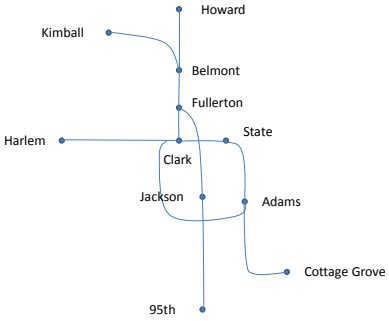
notions?

- adjacent, incident
- neighborhood
- connected
- path

Model

- What can we model with graphs?
- How can we extend the model?
  - directed/undirected
  - colored
  - labeled
  - weighted
- How do we represent them?
  - adjacency matrix, adjacency list

Example: CTA



Example: CTA

Trainstop

STOPID	NAME
1	Kimball
2	Howard
3	Belmont
4	Fullerton
5	Harlem
6	Clark
7	State
8	Adams
9	Jackson
10	Cottage Grove
11	95th Street

DirectedRoute

ROUTEID	NAME	DIRECTION
1	Red	North
2	Red	South
3	Green	North
4	Green	South
5	Brown	North
6	Brown	South

RouteDetail

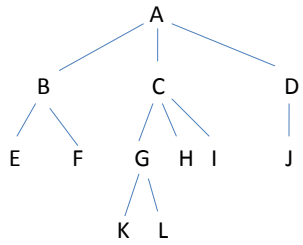
	ROUTEID	FROM_SID	TO_SID
1	1	11	
2	1	9	
3	1	4	
4	1	3	
5	2	2	
6	2	3	
7	2	4	
8	2	9	
9	3	10	
10	3	8	
11	3	7	
12	3	6	
13	4	5	
14	4	6	
15	4	7	
16	4	8	
17	5	8	
18	5	7	
19	5	6	
20	5	4	
21	5	3	
22	6	1	
23	6	3	
24	6	4	
25	6	6	

Processing Graphs

Graph Traversal

- breadth-first search
- depth-first search
- spanning tree/spanning forest
- can use with APIs or PL/SQL, often pre-implemented

TREES



ORACLE AND SQL DEVELOPER

Using SQLDeveloper

- First, set up new connection to computer running DBMS (see document)
  - cdmoracle.cdm.depaul.edu (140.192.39.58)
  - use your CSTCIS username, with initial password cti######## (your 7-digit Student ID)
  - To change password, ALTER USER *username* IDENTIFIED BY *newpassword* ; (…Execute)
- Double-click to open an existing connection

Getting SQL Developer

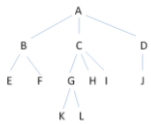
- Download SQL Developer
  - http://www.oracle.com/technology/software/products/sql/index.html
  - or http://ipd.cdm.depaul.edu/sqldeveloper/
- Install; gives visual interface to Oracle account

Example: create tree

```
drop table employee;
create table employee (
    emp_id              integer primary key,
    name                varchar2(2),
    supervisor_id       integer references employee
);

insert into employee values(1, 'A', null);
insert into employee values(2, 'B', 1);
insert into employee values(3, 'C', 1);
insert into employee values(4, 'D', 1);
insert into employee values(5, 'E', 2);
insert into employee values(6, 'F', 2);
insert into employee values(7, 'G', 3);
insert into employee values(8, 'H', 3);
insert into employee values(9, 'I', 3);
insert into employee values(10, 'J', 4);
insert into employee values(11, 'K', 7);
insert into employee values(12, 'L', 7);
```

Simple Examples




```
drop table employee;
create table employee (
    emp_id          integer
                primary key,
    name            varchar2(2),
    supervisor_id   integer
                references employee
);
```

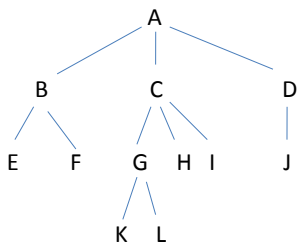
- Who is the boss?
- Who is K's supervisor?
- How many people do not supervise anybody?

Running SQL Scripts

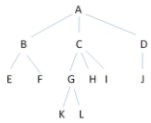
- A script is sequence of SQL commands stored in a file
- To run, type @*FullPath**File*.sql in center window, then click Run Script (F5)
- Any output will appear in bottom window (Results or Script Output)
- Click on Save icon to save output window contents to a text file (Click Clear icon first)

Networks in Oracle

TREE, again



TREES




```
drop table employee;
create table employee (
    emp_id          integer
                primary key,
    name            varchar2(2),
    supervisor_id   integer
                references employee
);
```

- Who is K's and I's first common supervisor?
- What is the shortest (official) route of communication between F and I?

Problem?

Solution Methods:
- connect by,
- Recursive SQL,
- PL/SQL, Java
- network model

CONNECT BY/START WITH

```
select name as emp_name,
       emp_id as employee,
       supervisor_id as sup_id
from employee
connect by prior emp_id = supervisor_id;
```

modify to display supervisor's name

```
select name as emp_name,
            emp_id as employee,
            supervisor_id as sup_id
from employee
connect by prior emp_id = supervisor_id;
start with name = 'A';
```

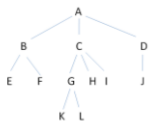
CONNECT BY/More features

- level pseudo-column
- use for indented display; *hint*: lpad(string, number)

```
select emp_id, level
from employee
connect by prior emp_id = supervisor_id
start with name = 'A';
```

- order employees by level
- get employees at lowest level

CONNECT BY/More features



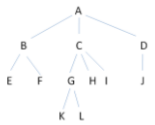
sys_connect_by_path: accumulate attributes along the traversal path
connect_by_root: find root attribute
connect_by_isleaf: true if node does not have a child

```
select name as emp_name,
   sys_connect_by_path(emp_id, ' ')
from employee
connect by prior emp_id = supervisor_id
start with name = 'A';

select emp_id, name, connect_by_root(emp_id)
from employee
connect by prior emp_id = supervisor_id;
```

modify to display supervisor's name

Tree Queries




```
drop table employee;
create table employee (
    emp_id          integer
                primary key,
    name            varchar2(2),
    supervisor_id   integer
                references employee
);
```

- Does K work for D?
- Who is K's and I's first common supervisor?
- What is the length of the shortest (official) route of communication between F and I?

Recursive SQL

```
with distance (from_id, to_id, dist) as
     (select emp_id, emp_id, 0 as dist
     from employee
  union all
     select E.supervisor_id, D.to_id, min(D.dist+1)
     from employee E, distance D
     where E.emp_id = D.from_id
     group by E.supervisor_id, D.to_id)
select from_id, to_id, dist from distance;
```

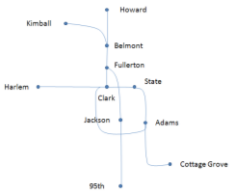
not supported by Oracle

PL/SQL

Goal: Calculate the distance between two vertices

- declare/begin/end, variables
  - calculate average employee level
  - output using dbms_output.put_line(…), use `set serveroutput on size 10000;'
- functions
  - create a function that converts polar to Euclidean coordinates
  - create function that test whether one employee is supervised by another
- control structures: if, loop
  - output names of employees with ids 4-10
- cursors
  - sum up all employee levels

Example: CTA



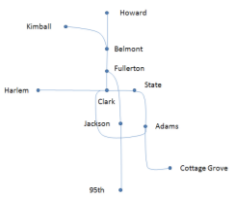
Trainstop

DirectedRoute

RouteDetail

- What stops are reachable from Howard?
- How many stops from Belmont to Jackson?
- What stops are reachable from Howard using red and brown line only?

Example: CTA



Trainstop

DirectedRoute

RouteDetail

- Write a PL/SQL function that counts how often we have to change trains for a given route
- Write a PL/SQL function that calculates the distance between two stops (use Haversine formula)

THE ORACLE NETWORK MODEL

Oracle Network Model

- Oracle has built-in network model for
  - nodes (node table)
  - link (link table)
  - paths (path table)
- can be created by hand (following conventions) or using built-in functions
- supports network analysis functions
- for more details, see Chapter 10 of *Pro Oracle Spatial* by Kothuri, Godfrink, Beinat
  available online at books 24x7
  (http://www.lib.depaul.edu/Find/resourceList.aspx?s=89)

Conventions

- node table:
  node_id number
- link table:
  link_id, start_node_id, end_node_id number
- path table:
  path_id, start_node_id, end_node_id number
- path link table:
  path_id, link_id, seq_no number

also: need to include geometry attributes in node, link and path, all of type sdo_geometry
names should be: node_name, link_name, etc.

Example: creating tables

```
create table ctanode (
  node_id number primary key,
  name  varchar2(30),
  location  sdo_geometry, -- the real geometry, we'll use our homemade version for now
  our_loc    gpoint); -- homemade geometry

create table ctalink (
  link_id  number primary key,
  start_node_id  number,
  end_node_id    number,
  distance       number,
  link_geom sdo_geometry
);

create table ctapath (
  path_id  number primary key,
  start_node_id  number,
  end_node_id    number,
  cost           number,
  path_geom      sdo_geometry
);

create table ctaplink (
  path_id  number,
  link_id  number,
  seq_no   number,
  constraint ctaplink_pk primary key (path_id, link_id)
);
```

Example: creating metadata

```
insert into user_sdo_network_metadata (network,
    network_category, geometry_type,
    no_of_hierarchy_levels, no_of_partitions,
    link_direction, node_table_name,
    node_geom_column, node_cost_column,
    link_table_name, link_geom_column,
    link_cost_column, path_table_name,
    path_geom_column, path_link_table_name)
values ('CTA_ROUTES', 'spatial', 'sdo_geometry', 1, 1,
    'directed', 'ctanode', 'location', null, 'ctalink',
    'link_geom', 'distance', 'ctapath', 'path_geom',
    'ctaplink');
```

Nearly done, but

- test that everything went fine:

```
select sdo_net.validate_network('CTA_ROUTES') from dual;
```

- to read network into memory (for analysis):

```
execute sdo_net_mem.network_manager.read_network('CTA_ROUTES',
    'FALSE');  -- false means read-only, true allows write
```

- to drop network from memory (not database):

```
execute sdo_net_mem.network_manager.drop_network('CTA_ROUTES');
```

- to write network into memory (after changes):

```
execute sdo_net_mem.network_manager.read_network('CTA_ROUTES');
```

Basic functions in sdo_net

- get_no_of_nodes()
  number of nodes in the network
- get_no_of_links()
  number of links in the network
- get_isolated_nodes()
  isolated nodes (no links) in network
- get_invalid_links()
- get_invalid_paths()

Basic function example

```
set serveroutput on size 10000;
declare
  x integer;
begin
  x := sdo_net.get_no_of_nodes('CTA_ROUTES');
  dbms_output.put_line('No of nodes: ' || x);
end;
```

More basic functions

- get_node_degree()
  number of incident links
- get_node_in_degree()
  number of links going out
- get_node_out_degree()
  number of links coming in
- get_in_links()
- get_out_links()

How to deal with a list: show all links out of Belmont

```
set serveroutput on size 10000;
declare
  x number;
link_ids sdo_number_array;
begin
  select node_id into x
  from ctanode
  where name = 'Belmont';
  link_ids := sdo_net.get_in_links('CTA_ROUTES', x);
  for i in link_ids.first..link_ids.last loop
       dbms_output.put_line(link_ids(i));
end loop;
end;
```

modify so we see the neighboring station instead of the links

Analysis functions

- sdo_net_mem.network_manager.shortest_path(<network>, <start_node>, <end_node>)
  - returns an integer identifying a path, information can be accessed using:
- sdo_net_mem.path.get_cost(<network>, <path>)
- sdo_net_mem.path.get_link_ids(<network>, <path>)
- sdo_net_mem.path.is_simple(<network>, <path>)
- sdo_net_mem.link.get_name(<link>)
- sdo_net_mem.link.get_cost(<link>)

• Write PL/SQL function that finds shortest path between two stations and lists the path taken.
• What about minimizing the number of changes?

Find close neighbors:

- sdo_net_mem.network_manager.nearest_neighbors(<network>, <start_node>, k)
  - returns the k closest neighbors
- sdo_net_mem.network_manager.within_cost(<network>, <start_node>, c)
  - returns neighbors within cost c

Comments:
• both functions use the cost attribute
• both functions do not return just the nodes, but arrays of paths to the nodes
  • can use sdo_net_mem.path.get_end_node_id

Finally, TSP

```
sdo_net_mem.network_manager.tsp_path(<network>, <nodes>, <is_closed>,
    <use_exact_cost>);
```

- Very restricted: list of nodes needs to be supplied
- is_closed: 'TRUE' forces return to first node