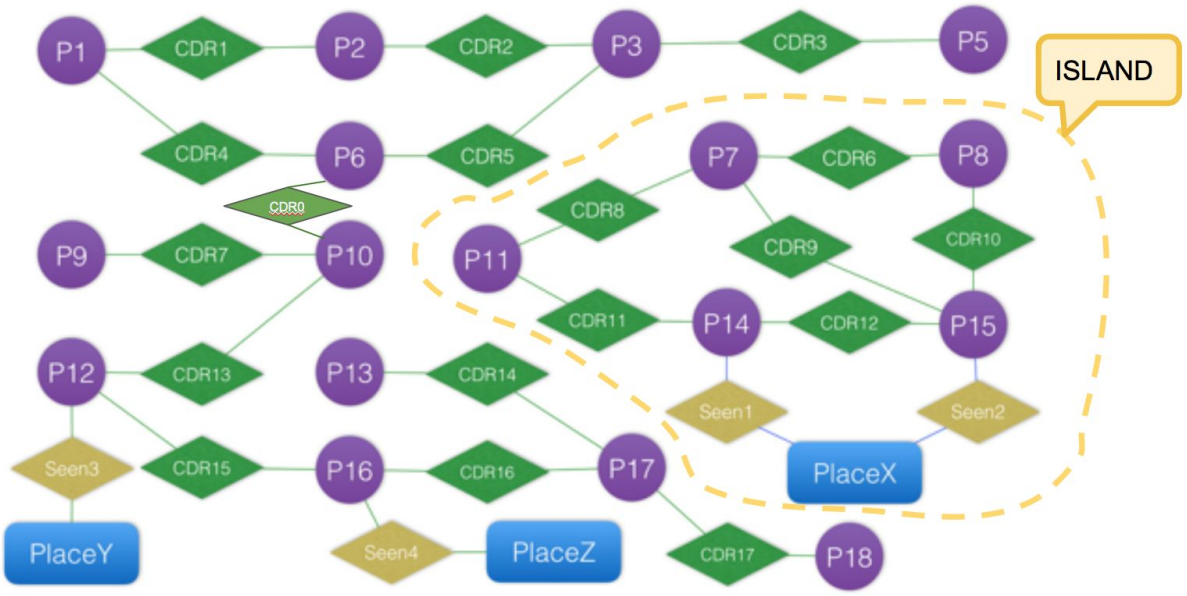


Objectivity ThingSpan Island Finding

1. Background

Islands in a graph are subgraphs that are not connected to any vertices outside of the ones in the subgraph. Islands can be important in many applications. In engineering an Island in the bill-of-materials (configuration) of a design cannot generally be a part of that item as it isn't connected toor included in anything. In crime enforcement an Island in a graph of telephone Call Detail Records may be symptomatic of a drug ring, terrorist cell or other criminal activity. In a social network it may represent a closely knit, dedicated community of followers.



2. Description of the Problem

Finding Islands in a large graph involves the examination of every vertex and edge in the graph to find subgraphs that aren't connected to the "external" graph. Current approaches, such as the Tarjin[1] closely connected graph algorithm, are very expensive in compute and I/O time costs and aren't very good at searching for Islands with less than a specified number of nodes. Looking for Islands with six or less nodes (people), for example, in the above situations is very inefficient with Tarjin.

[1] See https://en.wikipedia.org/wiki/Tarjan%27s_strongly_connected_components_algorithm

3. How is this problem being solved now and why isn't that acceptable?

All known solutions involve visiting all nodes in the graph and performing transitive closures (path navigation) until all subgraphs have been examined. Some allow the selection of a start node or randomly select nodes. Users can code the Tarjin algorithm or invent their own solution. Both techniques will be excessively slow at examining very large (billions of vertices) graphs.

4. Description of the Requested Features

4.1 Technique

We can improve the Tarjin algorithm, particularly for the small Island case, by combining the parallel, distributed processing capabilities of Apache Spark and our proprietary graph analytics platform, Objectivity ThingSpan. Objectivity has a patent filing underway that describes a completely new approach that is particularly good at finding Islands that have a number of vertices less than or equal to a chosen value.

4.2 Functionality

The user must be able to run a ThingSpan DO declarative graph and conventional data query that:

a) Specifies the graph or subgraph to be examined, denoted as S.
b) Identifies all Island subgraphs in S.
c) Optionally limits the total number of vertices in the Island subgraphs
d) Optionally limits the types of Vertex and Edge objects to be included in or excluded from the Island subgraphs.

5. What languages must support this capability?

- Java

6. Which platforms must be supported?

- Linux
- Solaris (later)

7. Do any competitors already have this feature?

- None are known.

8. **Customers who require this feature**

An improved Island finding algorithm and a scalable platform to run it on would be of interest to:

- The Intelligence Community.
- Network planning and configuration management specialists.
- Social network analysts.

9. Related Material To Be Developed

We will also need to provide:

- Quality Assurance material.
- Documentation
- Professional Services

10. Implementation Guidelines

10.1 Goals

There are three primary goals:

- Safe, high quality, low cost, fastest time to market.
- High performance, massive scalability and optimal physical storage efficiency.
- Minimal disruption to the Objectivity/DB kernel and ThingSpan DO API.

10.2 Priorities

Feature 4d) can be implemented as a second deliverable, namely:

- Optionally limit the types of Vertex and Edge objects to be included in or excluded from the Island subgraphs.

11. Additional Notes

This is a low risk project because most of the components exist and have been tested with one another, namely:

- Apache Spark.
- The ThingSpan graph database (which is based on Objectivity/DB, deployed within the Intelligence Community for almost two decades).
- The ThingSpan Spark Adaptor.
- The ThingSpan declarative graph query language, DO.
- All of the above will be running on standard platforms and the Amazon EC2 (or similar) Cloud.