Analysis of Tire Quality

Team Members: (Project Group 7)

Gabriel Nicolae-gxn031000

Navya Padala-nxp131930

Viswesh Muralidharan-vxm130730

Type of Project: Custom Project Type 3 - Using data from DATA-GOV and hosting our own SPARQL endpoint.

Project demo website: <http://www.utdallas.edu/~vxm130730/>

Project source code: <https://github.com/UTDSemWebSpring2015/analysis-of-tire-quality>

# Introduction

The purpose of our project is to help people make informed decisions when choosing tires for their vehicles. The main source of data for our project comes from the DATA-GOV website (Dataset 1353) that consists of tire quality parameters like Traction Performance, Temperature Resistance, Tread Wear etc, together with Brand and Tire Size information.

The end result of our project is a graphical data visualization tool for making easy to see how various tire lines compare to each other based on their quality parameters.

# Target Audience

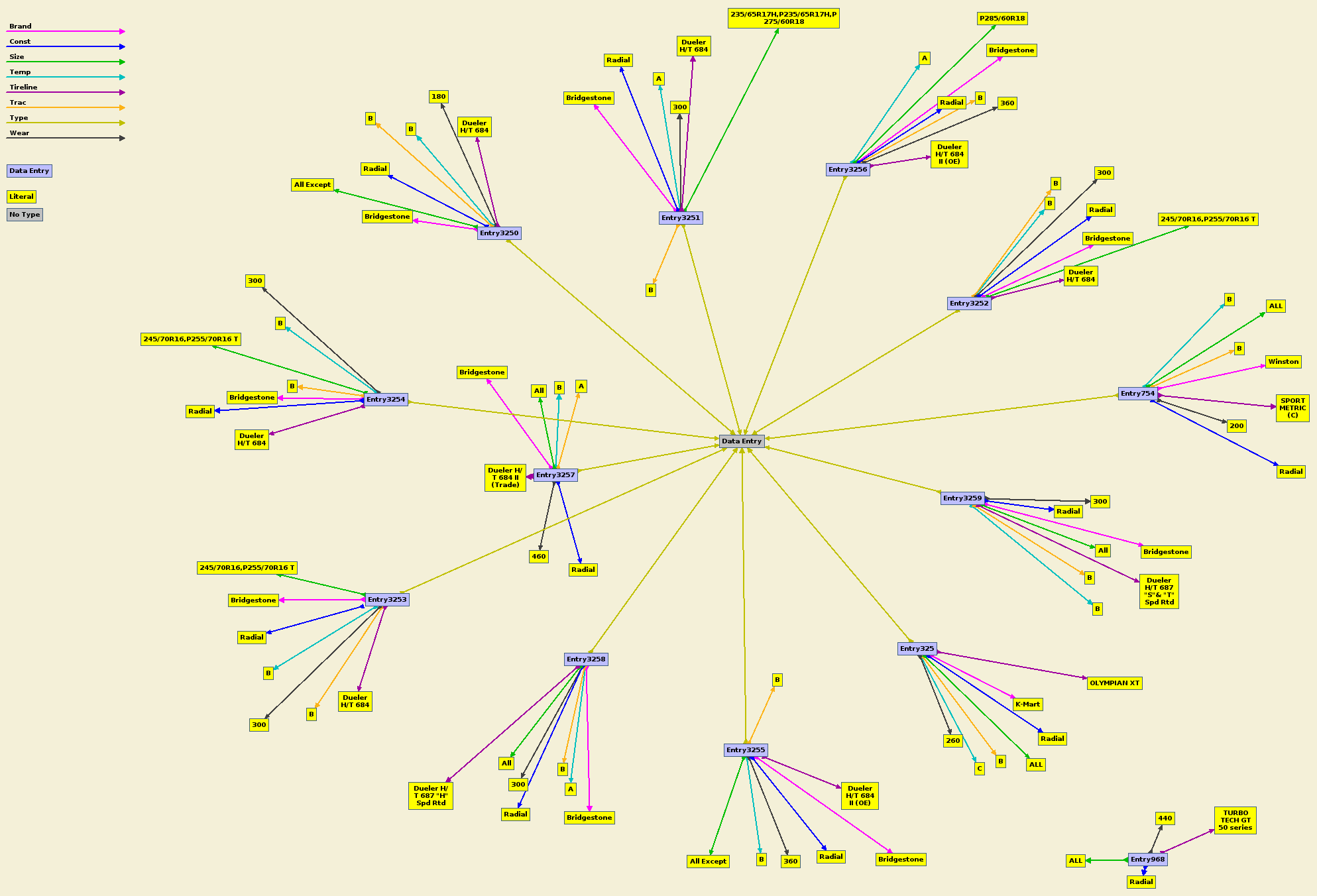
This project is targeted to the general consumers that are shopping for new or replacement tires for their vehicles. The data visualization helps the consumers trim down their buying options.

# Description of Data Sources

## Tires data

We have used two data sources for this project. The domain specific data source, Dataset 1353, was provided by DATA-GOV website (<http://data-gov.tw.rpi.edu/wiki/Dataset_1353>) and consists of 34,763 triplets to describe 4,345 data entries.

National Highway Traffic Safety Administration (NHTSA) has collected the data in order to assist consumers purchasing new vehicles or replacement tires. NHTSA has rated more than 4,200 lines of tires, including most used on passenger cars, minivans, SUVs and light pickup trucks using a grading system known as the Uniform Tire Quality Grading System (UTQGS).



## DBpedia

The second data source used in our project was DBpedia, a general-purpose data source. The DBpedia data was accessed from an existing SPARQL endpoint hosted by dbpedia.org: <http://dbpedia.org/sparql>.

DBpedia is a project aiming to extract structured content from the information created as part of the Wikipedia project and allows users to semantically query relationships and properties associated with Wikipedia resources, including links to other related datasets. At the time of writing this report DBpedia consisted of 883,644,431 triplets.

# Data Integration

Our first task in this project was to make our tires data (Dataset 1353) available for querying by hosting our own SPARQL endpoint. We have used the data provided by this endpoint as the main data source for our data visualization tool, which consists of HTML and JavaScript code.

To enhance the consumer experience during data visualization we have provided integration with DBpedia. We use DBpedia to collect and display to the user additional information about the tire brands, like brand logo or link to the Wikipedia page for a brand.

The integration between the two datasets is done in the JavaScript code by querying DBpedia to get additional information for tire brands.

Query example for selecting all tire brands from Dataset 1353:

SELECT DISTINCT ?brand

WHERE {

?subject <http://data-gov.tw.rpi.edu/vocab/p/1353/brand> ?brand

}

Query example for selecting additional brand data from DBpedia:

SELECT ?wikiLink WHERE

{

{

{

?subject a <http://dbpedia.org/class/yago/TireManufacturers>

}

UNION

{

?subject <http://purl.org/dc/terms/subject> <http://dbpedia.org/resource/Category:Tire_manufacturers>

}

}

{

{?subject foaf:name ?name .}

UNION

{?subject rdfs:label ?name .}

}

?subject foaf:isPrimaryTopicOf ?wikiLink.

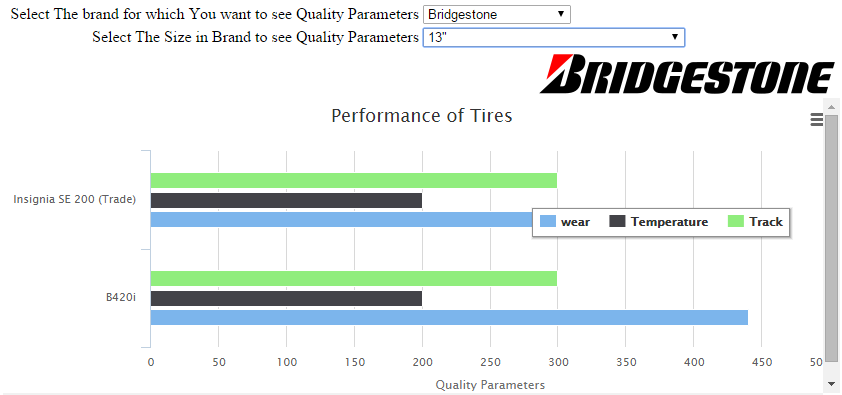
FILTER regex(?name, "Bridgestone", "i").

} LIMIT 1

The integration with DBpedia proved to be more difficult than initially anticipated because of missing pieces of information in DBpedia or data inconsistencies. For instance, some tire brands had no corresponding DBpedia entity which make it impossible for the tool to display additional information for that brand.

# Data product results

Project demo is hosted at <http://www.utdallas.edu/~vxm130730/>. The navigation consists of selecting a Tire Brand, followed by selecting a Tire Size from the list of available tire sizes produced by the brand manufacturer. The result will be a graph comparing the quality parameters for all Tire Lines that correspond to the Brand and Size selected.



5.1 Quality parameters for Bridgestone tires of size 13"

# Custom Project Justification

This is a custom project because we had to host our own SPARQL endpoint for Dataset 1353. We have used Fuseki1 (<http://jena.apache.org/documentation/serving_data/>) for hosting our SPARQL endpoint. The endpoint was hosted on Heroku (<https://www.heroku.com/>), a cloud application platform, and is available for querying at:

[https://analysis-of-tire-quality.herokuapp.com/tires/query?query=<yourquery](https://analysis-of-tire-quality.herokuapp.com/tires/query?query=%3cyourquery)>

Query example that selects five triplets from the dataset:

[https://analysis-of-tire-quality.herokuapp.com/tires/query?query=SELECT%20\*%20%7B?s%20?p%20?o%7D%20LIMIT%201](https://analysis-of-tire-quality.herokuapp.com/tires/query?query=SELECT%20*%20%7B?s%20?p%20?o%7D%20LIMIT%201)

Result:

{

"head": {

"vars": [ "s" , "p" , "o" ]

} ,

"results": {

"bindings": [

{

"s": { "type": "uri" , "value": "http://data-gov.tw.rpi.edu/raw/1353/data-1353.rdf#entry3994" } ,

"p": { "type": "uri" , "value": "http://data-gov.tw.rpi.edu/vocab/p/1353/wear" } ,

"o": { "type": "literal" , "value": "420" }

}

]

}

}

Note: Heroku apps go to sleep if they are not accessed for 10 minutes, so you might get a timeout or slow response the first time you try to access the service. Subsequent queries less than 10 minutes apart should be served faster.

# Summary

This project provides a graphical visualization tool for comparing quality parameters of various tires, which will help people decide which tries to buy for their car.

We have hosted our own SPARQL endpoint for querying data from Dataset 1353, and we have provided a level of integration with DBpedia data. The querying, data integration and visualization were implemented using HTML and JavaScript.