Improving text search selectivity

(a Google Summer of Code project)
Every good piece of software starts by scratching a developer's personal itch.

- Eric S. Raymond
explain select * from docs where tsvector @@ to_tsquery('hippos');

QUERY PLAN

Seq Scan on docs  (cost=0.00..1420.08 rows=11 width=71583024)
  Filter: (tsvector @@ to_tsquery('hippos'::text))
explain select * from docs where tsvector @@ to_tsquery('hippos');

QUERY PLAN

Seq Scan on docs  (cost=0.00..1420.08 rows=11 width=71583024)
  Filter: (tsvector @@ to_tsquery('hippos':::text))

explain select * from docs where tsvector @@ to_tsquery('dogs');

QUERY PLAN

Seq Scan on docs  (cost=0.00..1420.08 rows=11 width=71583024)
  Filter: (tsvector @@ to_tsquery('dogs':::text))
= explain select * from docs where tsvector @@ to_tsquery('hippos');

**QUERY PLAN**

Seq Scan on docs  (cost=0.00..1420.08 rows=11 width=71583024)
  Filter: (tsvector @@ to_tsquery('hippos '::text))

= explain select * from docs where tsvector @@ to_tsquery('dogs');

**QUERY PLAN**

Seq Scan on docs  (cost=0.00..1420.08 rows=11 width=71583024)
  Filter: (tsvector @@ to_tsquery('dogs '::text))

= explain select * from docs where tsvector @@ to_tsquery('foo & quuz');

**QUERY PLAN**

Seq Scan on docs  (cost=0.00..1420.08 rows=11 width=71583024)
  Filter: (tsvector @@ to_tsquery('foo & quuz '::text))
```sql
=# explain select * from docs where tsvector @@ to_tsquery('hippos');

QUERY PLAN
-----------------------------------------------
Seq Scan on docs (cost=0.00..1420.08 rows=11 width=71583024)
  Filter: (tsvector @@ to_tsquery('hippos'::text))

=# explain select * from docs where tsvector @@ to_tsquery('dogs');

QUERY PLAN
-----------------------------------------------
Seq Scan on docs (cost=0.00..1420.08 rows=11 width=71583024)
  Filter: (tsvector @@ to_tsquery('dogs'::text))

=# explain select * from docs where tsvector @@ to_tsquery('foo & quuz');

QUERY PLAN
-----------------------------------------------
Seq Scan on docs (cost=0.00..1420.08 rows=11 width=71583024)
  Filter: (tsvector @@ to_tsquery('foo & quuz'::text))
```
• PostgreSQL assumes a fixed selectivity estimate for the @@ operator

• Obviously, this leads to some very suboptimal plans

• Less obviously, it's not easily fixed

• This GSoC project tries to do something about it
- Type-specific ANALYZE functions
- Default fallback routine
- As of now, there are no type-specific functions
- Tsvectors are actually different
- Determine most common lexemes, instead of most common values
Bogus contsel function

Custom selectivity functions

Need to know how many rows contain a given lexeme

Simple top-N, but may prove sufficient

See Zipf's law
• Easy to implement through standard interfaces
• Completely implementable in userspace (!)
• PostgreSQL rocks