



The Kalmanovitz Library and
The Center for Knowledge Management

ELink E-Utilities

Gilberto da Gente
Bioinformatics Specialist
May 22th 2008

University of California, San Francisco



UCSF

Overview

- Accessing Entrez links
 - Hard links between databases
 - Computational links within a database
 - Filtering according to the existence of links

Entrez Links for GI 4680720

1: [M17755](#)

Homo sapiens thyroid peroxidase (TPO) mRNA, complete cds
gi4680720|gb|M17755.2|HUMTPOC[4680720]

Gene annotation based on M17755

Full text online articles about M17755

All polymorphisms in the TPO gene

DNA/RNA sequences similar to M17755

Graphical view of TPO gene annotation

Human phenotypes involving TPO

Microarray datasets for M17755

Protein translation of M17755

Literature abstracts about M17755

Sequence polymorphisms in M17755

Source organism of M17755

STS markers in the TPO gene

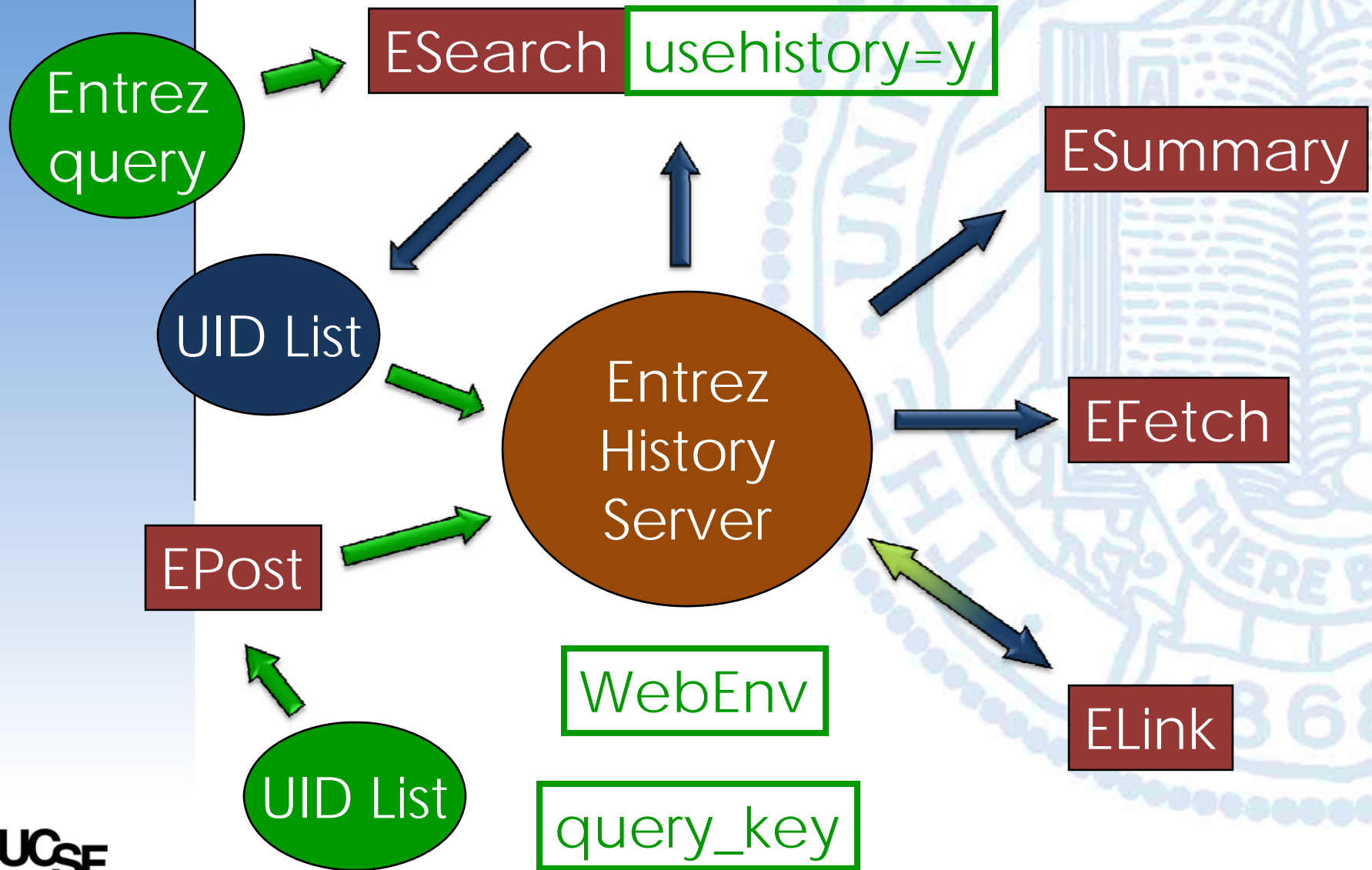
TPO links beyond NCBI

Links

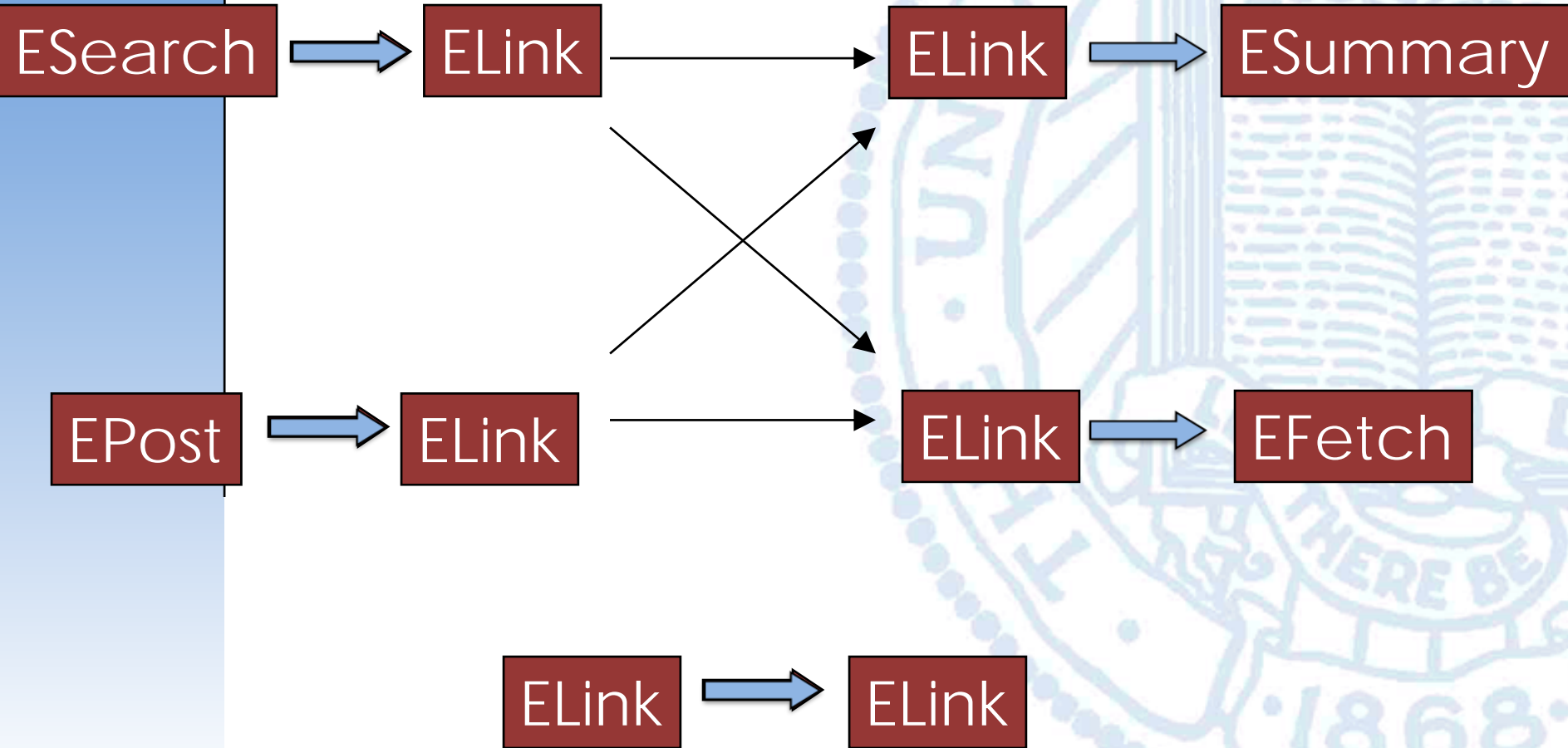
Links

- Gene
- Full text in PMC
- GeneView in dbSNP
- Related Sequences
- Map Viewer
- OMIM
- GEO Profiles
- Protein
- PubMed
- SNP
- Taxonomy
- UniGene
- LinkOut

The Big Picture

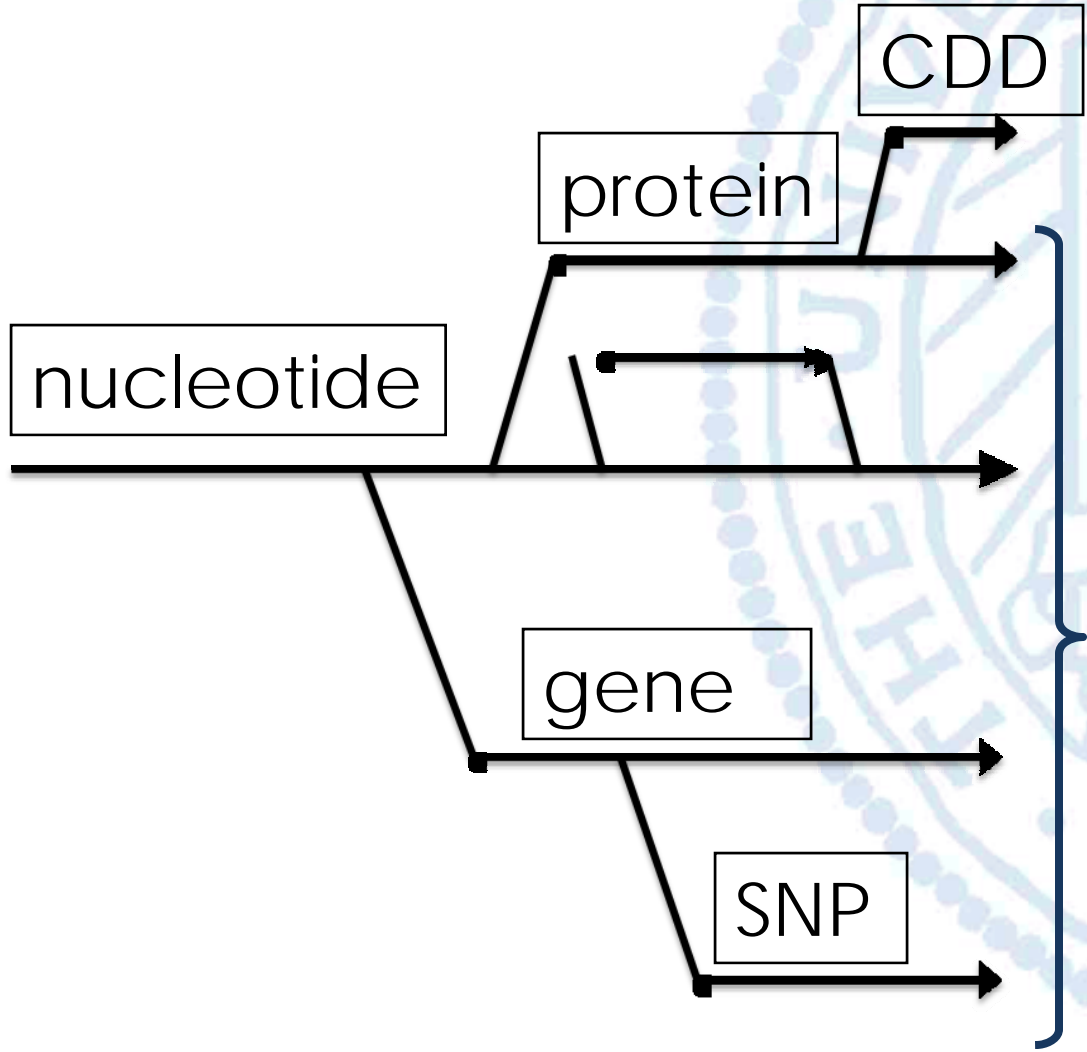


Basic ELink Pipelines



ELink: Forks in the Road

Entrez query



DocSums

Formatted Data

ELink

Retrieves UID(s) in database B linked to a set of UID(s) in database A

Why use it?

- To find related data in another database
- To find neighbors within a database

INPUT

dbfrom

Entrez database to link *from*

db

Entrez database(s) to link *to*; (can be a list)

id

List of UID(s)

cmd

ELink command mode (default = neighbor)

BASE/

elink.fcgi?

dbfrom=protein&db=pubmed&id=148762980

OUTPUT

XML

Set(s) of linked UID(s)

ELink Output

```
<eLinkResult>
<LinkSet>
  <DbFrom>protein</DbFrom>
  <IdList>
    <Id>148762980</Id>
  </IdList>
  <LinkSetDb>
    <DbTo>pubmed</DbTo>
    <LinkName>protein_pubmed</LinkName>
    <Link>
      <Id>17012248</Id>
    </Link>
  </LinkSetDb>
  <LinkSetDb>
    <DbTo>pubmed</DbTo>
    <LinkName>protein_pubmed_refseq</LinkName>
    <Link>
      <Id>17012248</Id>
    </Link>
    <Link>
      <Id>12975309</Id>
    </Link>
  </LinkSetDb>
</LinkSet>
</eLinkResult>
```

&cmd=neighbor

Returns **all** linked UIDs
(can be thousands...!)

&cmd=neighbor_history

Returns only WebEnv
and query keys

```
<eLinkResult>
<LinkSet>
  <DbFrom>protein</DbFrom>
  <IdList>
    <Id>148762980</Id>
  </IdList>
  <LinkSetDbHistory>
    <DbTo>pubmed</DbTo>
    <LinkName>protein_pubmed</LinkName>
    <QueryKey>9</QueryKey>
  </LinkSetDbHistory>
  <LinkSetDbHistory>
    <DbTo>pubmed</DbTo>
    <LinkName>protein_pubmed_refseq</LinkName>
    <QueryKey>8</QueryKey>
  </LinkSetDbHistory>
  <WebEnv>OkQMgnlRqL2spNJdxipACnRX9jjrugeTLHLcUFnGzGN8D6
</WebEnv>
</LinkSet>
</eLinkResult>
```

Link Names

linkname

Name of link to retrieve (if omitted, all link names are retrieved)

gene_protein

Links from gene to protein

protein_gene

Links from protein to gene

gene_snp

Links from gene to snp

gene_snp_genegenotype

Links from gene to snps that have genotype data

Link names for a given call are given in the ELink XML output

All possible link names for a database are given by EInfo

Specifying a Link Name

```
dbfrom=protein&db=pubmed&id=148762980
```



```
<eLinkResult>
<LinkSet>
  <DbFrom>protein</DbFrom>
  <IdList>
    <Id>148762980</Id>
  </IdList>
  <LinkSetDb>
    <DbTo>pubmed</DbTo>
    <LinkName>protein_pubmed</LinkName>
    <Link>
      <Id>17012248</Id>
    </Link>
  </LinkSetDb>
  <LinkSetDb>
    <DbTo>pubmed</DbTo>
    <LinkName>protein_pubmed_refseq</LinkName>
    <Link>
      <Id>17012248</Id>
    </Link>
    <Link>
      <Id>12975309</Id>
    </Link>
  </LinkSetDb>
</LinkSet>
</eLinkResult>
```

```
dbfrom=protein&db=pubmed
&id=148762980&linkname=protein_pubmed
```



```
<eLinkResult>
<LinkSet>
  <DbFrom>protein</DbFrom>
  <IdList>
    <Id>148762980</Id>
  </IdList>
  <LinkSetDb>
    <DbTo>pubmed</DbTo>
    <LinkName>protein_pubmed</LinkName>
    <Link>
      <Id>17012248</Id>
    </Link>
  </LinkSetDb>
</LinkSet>
</eLinkResult>
```

Computational Neighbors in ELink

dbfrom

=

db

Retrieves UID's linked to other UID's in the same database **and** their similarity scores

term

Entrez query that ELink uses to limit the set of neighbors

BASE/

elink.fcgi?

dbfrom=protein&db=protein&id=15718680&term=lemurs[orgn]

Supported databases:

pubmed	cdd
nucleotide	geo
protein	gds
domains	

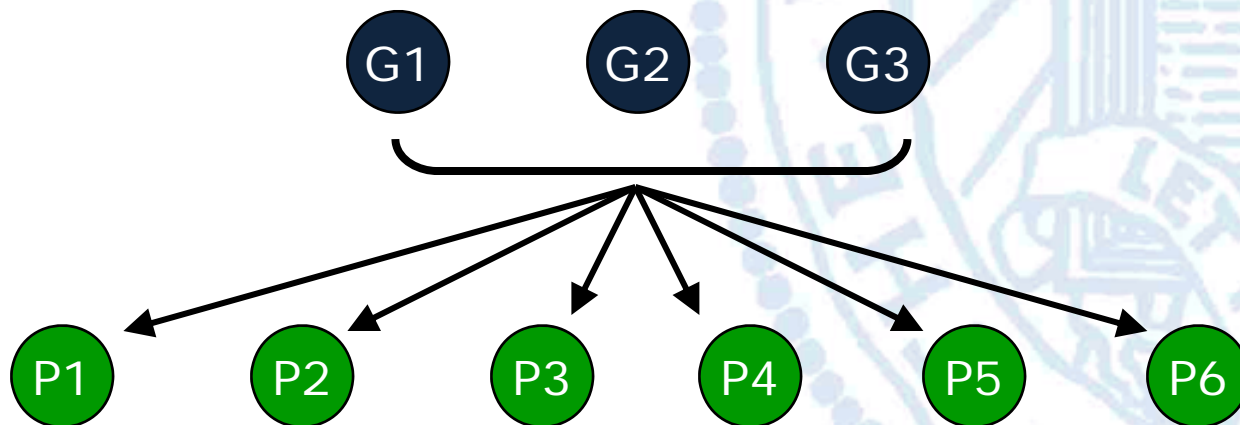
Link Results

```
<eLinkResult>
  <LinkSet>
    <DbFrom>protein</DbFrom>
    <IdList>
      <Id>15718680</Id>
    </IdList>
    <LinkSetDb>
      <DbTo>protein</DbTo>
      <LinkName>protein_protein</LinkName>
      <Link>
        <Id>15718680</Id>
        <Score>2147483647</Score>
      </Link>
      <Link>
        <Id>68270972</Id>
        <Score>458</Score>
      </Link>
      <Link>
        <Id>86211625</Id>
        <Score>458</Score>
      </Link>
      <Link>
        <Id>118572785</Id>
        <Score>458</Score>
      </Link>
      <Link>
        <Id>148877256</Id>
        <Score>235</Score>
      </Link>
      <Link>
        <Id>134093096</Id>
        <Score>235</Score>
      </Link>
    </LinkSetDb>
  </LinkSet>
</eLinkResult>
```

Self hit

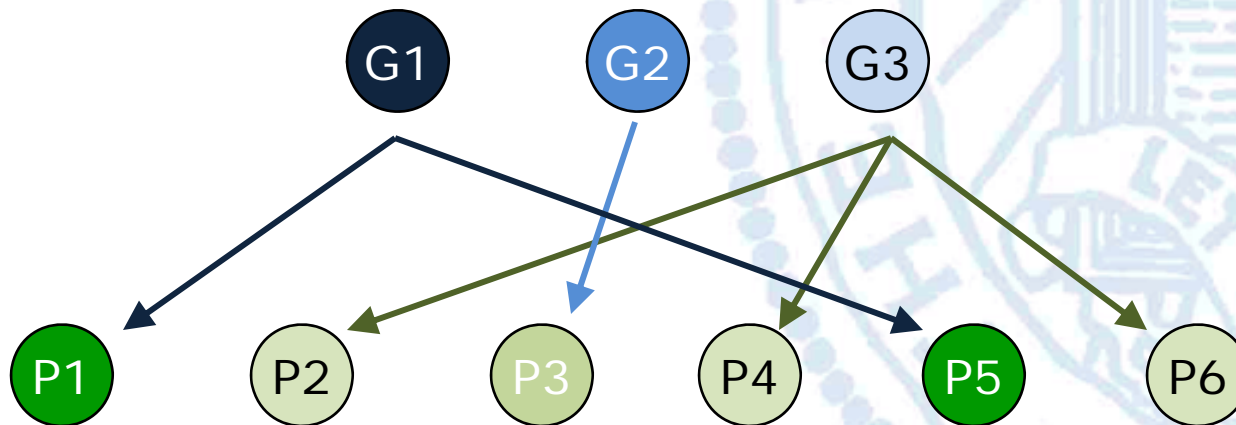
Passing One UID Set to ELink

```
dbfrom=gene&db=protein&id=G1,G2,G3
```



Passing Multiple UID Sets to ELink

```
dbfrom=gene&db=protein&id=G1&id=G2&id=G3
```



Three Paths to Links in Entrez

Find links between a set of protein sequences and their SNPs

I `elink_batch_to`

`dbfrom=protein&db=snp&id=P1,P2`

Finds **all SNPs** linked to the set (P1, P2)

II `elink_by_id_to`

`dbfrom=protein&db=snp&id=P1&id=P2`

Finds **separate lists** of SNPs linked to P1 and P2

III `esearch`

`db=protein&term=protein+snp[filter]`

Finds **all proteins** that have links to SNPs

sub elink_batch_to

The Problem: Upper limit on the input set size for ELink
The Solution: Break up the input set into smaller batches

sub elink_batch_to

Finds one set of links for the entire set of input
UIDs (batch = 25000)

```
%links = elink_batch_to($db,%params);
```

1. Break up input set into batches of 25000
If no linkname is specified, then for each linkname...
2. Find links for each batch
3. Non-redundify the resulting set of links
4. Post the non-redundant set onto the History
5. Return (db, WebEnv, query_key)

sub elink_batch_to

Find phenotypes and SNPs linked to a posted set of Gene IDs

becomes
&dbfrom

```
$params{db}  
$params{query_key}  
$params{WebEnv}
```

From esearch or
epost_file

```
%links = elink_batch_to(\omim,snp',%params);
```

```
$links{\gene_omim}{\db'} = \omim'
```

```
$links{\gene_omim}{\query_key'} = query key for OMIM UIDs
```

```
$links{\gene_omim}{\WebEnv'} = WebEnv for OMIM UIDs
```

```
$links{\gene_snp}{\db'} = \snp'
```

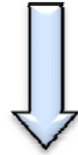
```
$links{\gene_snp}{\query_key'} = query key for SNP UIDs
```

```
$links{\gene_snp}{\WebEnv'} = WebEnv for SNP UIDs
```

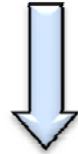
sub extract_links

Extracts data packet for a single linkname from %links

```
$links{'gene_omim'}{'db'} = 'omim'  
$links{'gene_omim'}{'query_key'} = '3'  
$links{'gene_omim'}{'WebEnv'} = 'J3f8dDFG83k'
```



```
%params = extract_links('gene_omim',%links);
```



```
$params{'db'} = 'omim'  
$params{'query_key'} = '3'  
$params{'WebEnv'} = 'J3f8dDFG83k'
```

Specifying a Link Name

If given a linkname, `elink_batch_to` will produce a simple hash

Find the *gene_omim* links for a posted set of Gene IDs

```
$params{db}
$params{query_key}
$params{WebEnv}
$params{linkname} = 'gene_omim';
%links = elink_batch_to('omim', %params);
```

From `esearch` or `epost_file`

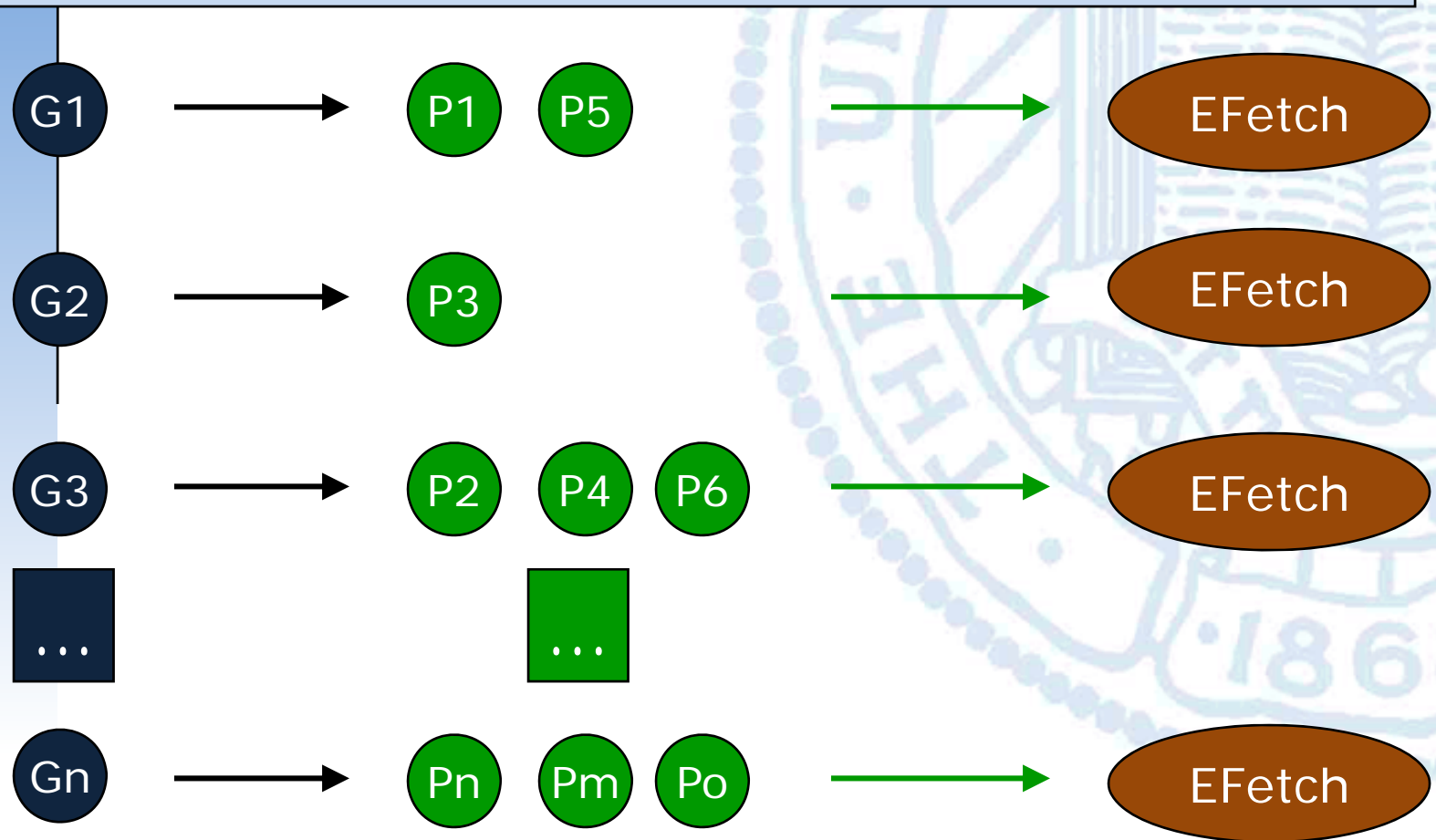


```
$links{'db'} = 'omim'
$links{'query_key'} = query key for OMIM UIDs
$links{'WebEnv'} = WebEnv for OMIM UIDs
```

Handling Links 'By Id'

1000 input genes produces 1000 separate sets of linked UIDs.

Downloading these requires 1000 EFetch URLs!



ELink By Id

The Problem: Upper limit on the input set size for ELink
The Solution: Break up the input set into smaller batches

```
sub elink_by_id_to
```

Finds separate sets of links for each input UID (batch = 5000)

```
%results = elink_by_id_to($db,%params);
```

1. Break up input set into batches of 5000
- If no linkname is specified, then for each linkname...***
2. Find links for each UID in the batch
3. Writes each set of links to an index file
4. Posts the non-redundant set of links **for all UIDs** onto the History
5. Return (db, WebEnv, query_key) for the entire set of links

Index Files

Each line represents one input UID

UID in
dbfrom

:

List of linked UIDs in db

```
2923:398564,318185,8940,31565,174978,817760,287930,416771,1274415,  
21976  
3394:454488,35236,303194,841933,416518,21975,444125,829248,836513,  
489537,3044971,7156,416517,1279455,831239,176631  
830:2708055,21973,855636,2892516,35225,2756048,1279885,811898,  
3033284,821972,2540541,395570,815856,454651,480525,174825,7153,  
3062917,2684328  
68155:460232,7155,189967,361100,396917,185157,85339,477044,360243,  
395966,185188,282097,324695,449876,21974,398512,323733  
4237:37189,10210,334469,818496,66434,106021,819689,324197,362501,  
481577
```

Score Files



```
2923:1412,1323,1218,1090,982,817,689,416,127,109
3394:4544,3522,3031,841,416,219,125,121,101,101,101,101,99,98,98,95
830:2708,2197,1788,1650,1501,1312,1312,1288,1258,1182,1122,1098,1055,1
012,989,825,715,676,332
68155:4602,4522,3821,3611,3607,3321,3039,2879,2443,2066,1851,1821,179
9,1744,1629,1522,1022
4237:371,322,322,311,289,221,196,192,187,157
```

UID in
dbfrom

List of similarity
scores

The entire set of linked UIDs in db is stored at
(query_key, WebEnv)

Using Index Files

Find nucleotides linked to 200 Gene IDs (assume >5 sec/URL)

Without index file: 1 elink URL and 200 efetch URLs (>1000 sec)

With index file: 1 elink URL and 1 efetch URL (>10 sec)

To download the linked DocSums or data, download the entire set using (query_key, WebEnv) and then use the **index file** and **read_index** to parse the desired records from the large set.

```
sub read_index
```

Reads index files and score files produced by elink_by_id_to

```
%index = read_index($filename);
```

```
$index{$id} – comma-delimited list of UIDs linked to $id OR scores
```

sub elink_by_id_to

Find separate sets of phenotypes and SNPs for each Gene ID

becomes
&dbfrom

```
$params{db}  
$params{query_key}  
$params{WebEnv}
```

From esearch or
epost_file

```
%links = elink_by_id_to('omim',%params);
```

Index file

Score file (if db = dbfrom)

```
$links{'gene_omim'}{'db'} = database of linked OMIM UIDs  
$links{'gene_omim'}{'query_key'} = query key for ALL linked OMIM UIDs  
$links{'gene_omim'}{'WebEnv'} = WebEnv for ALL linked OMIM UIDs  
$links{'gene_omim'}{'linkfile'} = filename of index file for OMIM links  
$links{'gene_snp'}{'db'} = database of linked SNP UIDs  
$links{'gene_snp'}{'query_key'} = query key for ALL linked SNP UIDs  
$links{'gene_snp'}{'WebEnv'} = WebEnv for ALL linked SNP UIDs  
$links{'gene_snp'}{'linkfile'} = filename of index file for SNP links
```

Specifying a Link Name

Find separate sets of *gene_omim* links for each Gene ID

```
$params{db}  
$params{query_key}  
$params{WebEnv}  
$params{linkname} = 'gene_omim';  
%links = elink_by_id_to('omim',%params);
```



+

Index file

Score file (if db = dbfrom)

```
$links{'db'} = database of linked OMIM UIDs  
$links{'query_key'} = query key for ALL linked OMIM UIDs  
$links{'WebEnv'} = WebEnv for ALL linked OMIM UIDs  
$links{'linkfile'} = filename of index file
```

The Pair Library: Part II

Link Branching

ESearch → ELink

search_link_batch.pl

search_link_by_id.pl

EPost → ELink

post_link_batch.pl

post_link_by_id.pl

ELink → ELink

link_link_batch.pl

link_link_by_id.pl

Link Processing

ELink → ESummary

link_batch_summary.pl

link_by_id_summary.pl

ELink → EFetch

link_batch_fetch.pl

link_by_id_fetch.pl