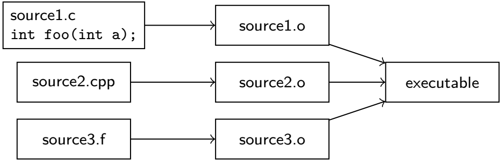
Incremental LTO in GCC

Michal Jireš

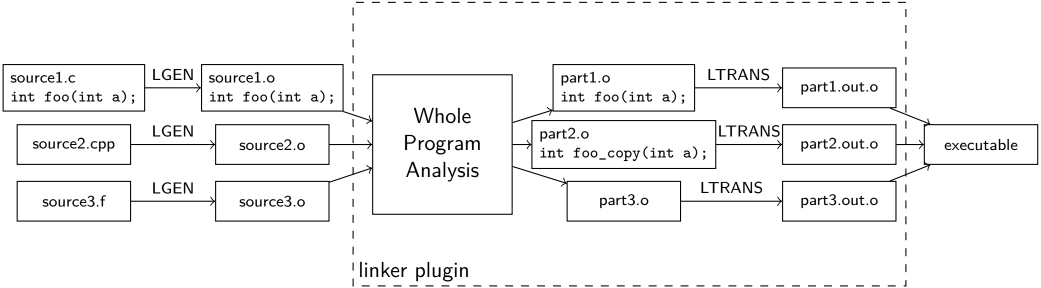
SUSE Labs

February 1, 2025

Standard compilation

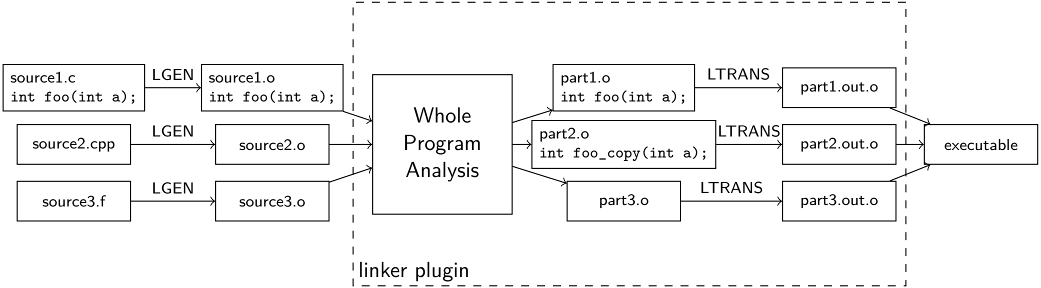


LTO

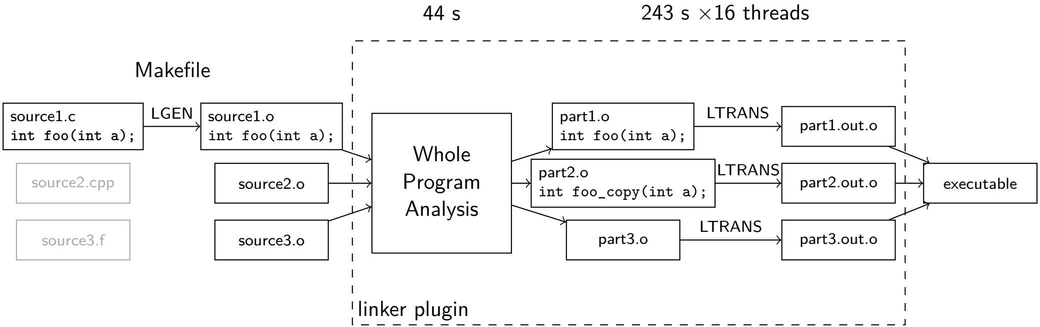

source1.c
int foo(int a);
LGEN
source1.o
int foo(int a);
source2.cpp
LGEN
source2.o
source3.f
LGEN
source3.o
Whole Program Analysis
part1.o
int foo(int a);
LTRANS
part1.out.o
part2.o
int foo_copy(int a);
LTRANS
part2.out.o
part3.o
LTRANS
part3.out.o
executable
linker plugin

LTO

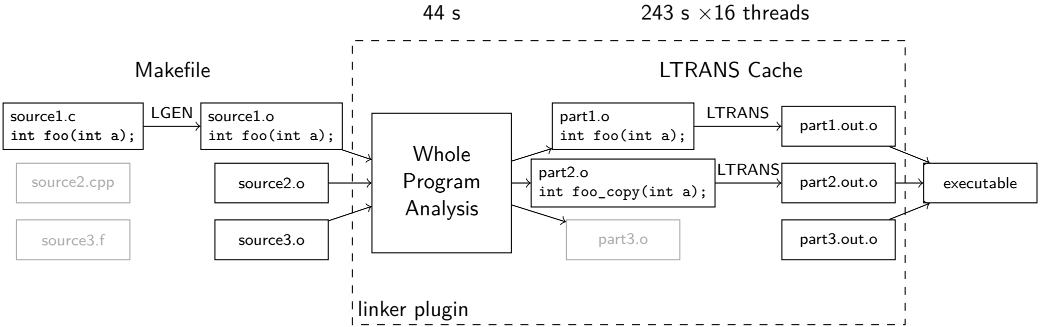
…-O2 -flto -o cc1: 44 s 243 s ×16 threads



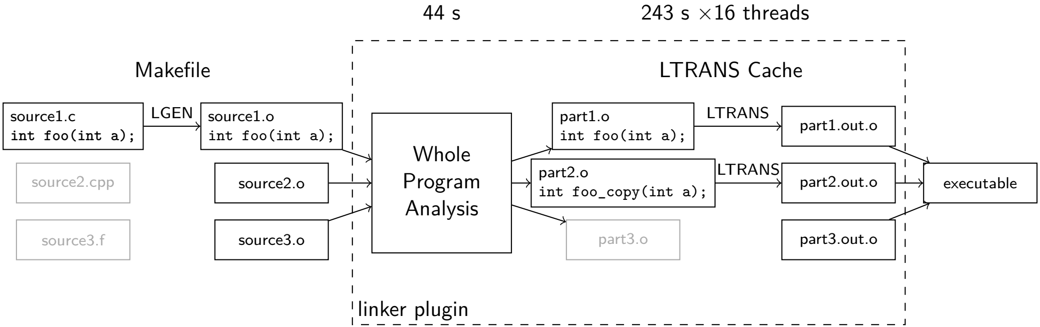
LTO


44 s
243 s ×16 threads
Makefile
source1.c
int foo(int a);
LGEN
source1.o
int foo(int a);
source2.cpp
source2.o
source3.f
source3.o
Whole Program Analysis
part1.o
int foo(int a);
LTRANS
part1.out.o
part2.o
int foo_copy(int a);
LTRANS
part2.out.o
part3.o
LTRANS
part3.out.o
executable
linker plugin

Incremental LTO



Incremental LTO



- Cache is useless if we never produce the same partition again.

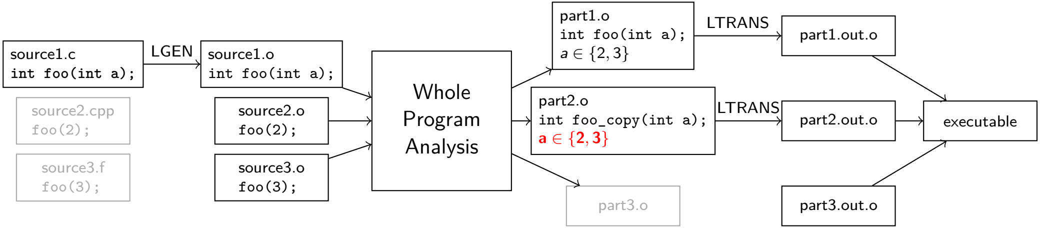
Global counters

- Counters that increment with each new item.

```
01 int random () {
02 return 4;
03 }
04 int foo () {
05 return 42;
06 }
```

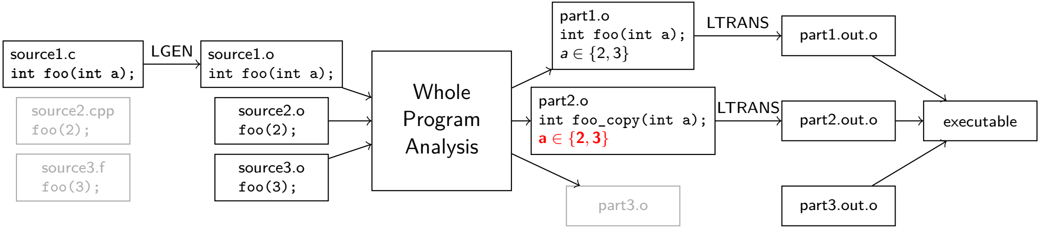
```
01 int random () {
02 static int r = 4;
03 return r++;
04 }
05 int foo () {
06 return 42;
07 }
```

Propagation of unused info



- Inline copies do not need to know all possible argument values.

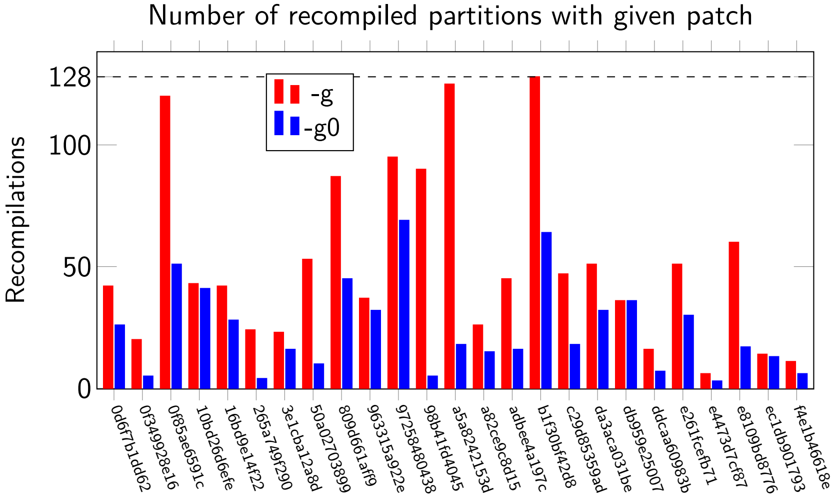
Propagation of unused info



- Inline copies do not need to know all possible argument values.
- `debug_function(__LINE__);`

Incremental LTO in GCC15

Self compiling -O2 cc1, with random patches representing small changes



Number of recompiled partitions with given patch

Relevant flags

- `-flto-incremental=/tmp/cache/`
    - filelocked → safe to use in parallel
- `-flto-partition=cache`
- `--param=lto-partitions=256` (defaults to 128)

Q&A