Implication Conditions as Variability Model

The *undertaker* approach to `KConfig`

Reinhard Tartler, Christian Dietrich & Christoph Egger

Department of Computer Science 4

Distributed Systems and Operating Systems

University of Erlangen-Nuremberg

March 9, 2010

supported by DFG

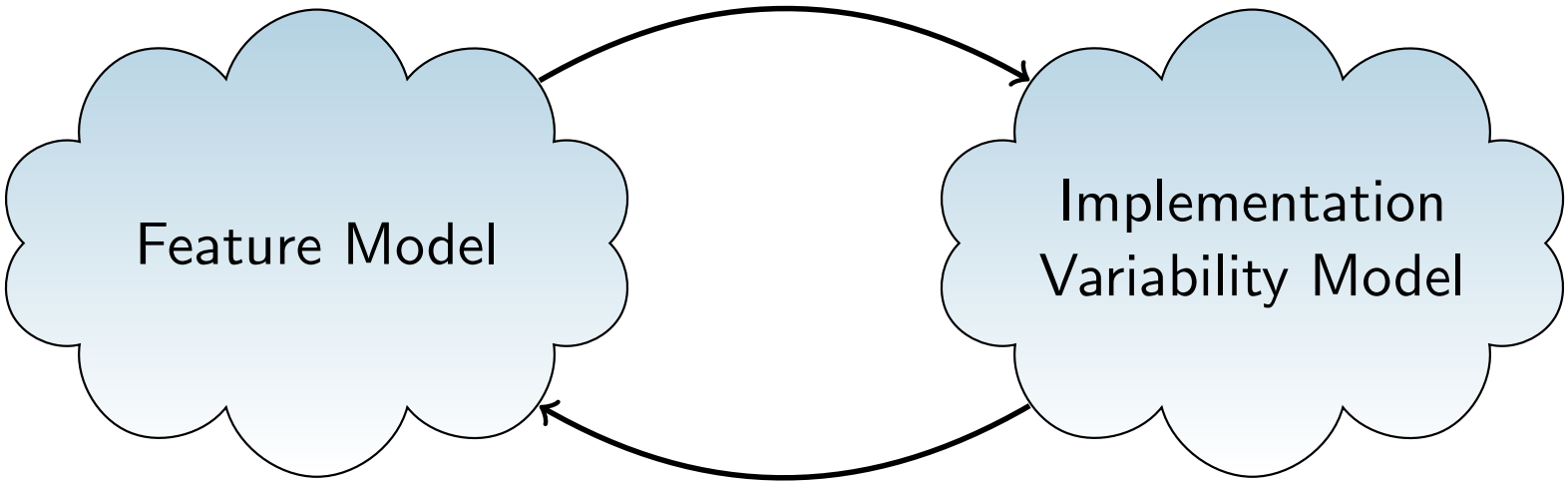
About us

Members of the DFG VAMOS Research project:

- Reinhard Tartler
- Christian Dietrich
- Christoph Egger

Motivation and Context

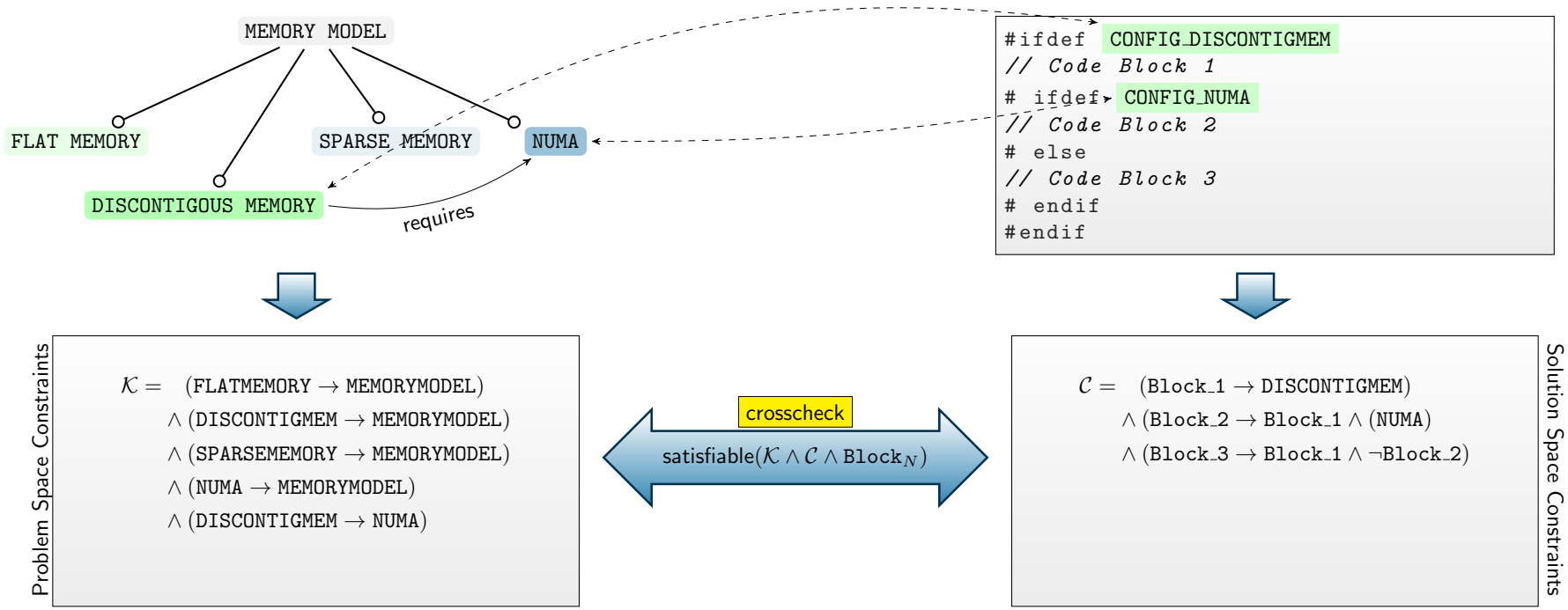
- Kconfig: The Feature Model of Linux [OPLC-OSSPL 2007]
- Linux with its over 10000 features is a perfect candidate for variability analysis [EuroSys 2011]
- Implementation is developed independently from the Feature Model



Big Source of Bugs!

The undertaker approach

Existing, FOSD agnostic, software is analyzed for variation points:



undertaker extracts propositional formulas

Classification of Configuration Defects

```
diff --git a/kernel/smp.c b/kernel/smp.c
--- a/kernel/smp.c
+++ b/kernel/smp.c

-#ifdef CONFIG_CPU_HOTPLUG
+#ifdef CONFIG_HOTPLUG_CPU
```

Patch for a symbolic defect

Classification of Configuration Defects

```
diff --git a/kernel/smp.c b/kernel/smp.c
--- a/kernel/smp.c
+++ b/kernel/smp.c

-#ifdef CONFIG_CPU_HOTPLUG
+#ifdef CONFIG_HOTPLUG_CPU
```

Patch for a symbolic defect

```
diff --git a/arch/x86/include/asm/mmzone_32.h
           b/arch/x86/include/asm/mmzone_32.h
--- a/arch/x86/include/asm/mmzone_32.h
+++ b/arch/x86/include/asm/mmzone_32.h
@@ -61,11 +61,7 @@ extern s8
  physnode_map[];

  static inline int pfn_to_nid(unsigned long pfn) {
-#ifdef CONFIG_NUMA
    return((int) physnode_map[(pfn) / PAGES_PER_ELEMENT]);
-#else
-   return 0;
-#endif
  }

  /*
```

Patch for a logical defect

Our results so far from the Dead Block Analysis

subsystem	#ifdefs	logic	symbolic	total
arch/	33757	345	581	926
drivers/	32695	88	648	736
fs/	3000	4	13	17
include/	7241	6	11	17
kernel/	1412	7	2	9
mm/	555	0	1	1
net/	2731	1	49	50
sound/	3246	5	10	15
virt/	53	0	0	0
other subsystems	601	4	1	5
$\sum$	85291	460	1316	1776
fix proposed		150 (1)	214 (22)	364 (23)
confirmed defect		38 (1)	116 (20)	154 (21)
confirmed rule-violation		88 (0)	21 (2)	109 (2)
pending		24 (0)	77 (0)	101 (0)

Conclusions

- Undertaker finds and buries real problems
  - Over 100 patches submitted!
  - has already received rave reviews by kernel developers and even the Linux Magazine [LM 04/11]
- Not really Linux specific
  - CPP annotated code is pretty common in e.g. system software
  - Configuration Models are often hard to access

  $\Rightarrow$ Need to write variability extractors for other projects
- Underapproximations are totally sufficient
- The better the model, the better results, obviously

Conclusions (?)

- Undertaker finds and buries real problems
  - Over 100 patches submitted!
  - has already received rave reviews by kernel developers and even the Linux Magazine [LM 04/11]
- Not really Linux specific
  - CPP annotated code is pretty common in e.g. system software
  - Configuration Models are often hard to access

  $\Rightarrow$ Need to write variability extractors for other projects
- Underapproximations are totally sufficient
- The better the model, the better results, obviously

Variability extraction for Kconfig

- Kconfig is a tool and language to describe variability
- Used by both developers and users to configure a specific variant
- Text and graphical front ends are available
- Features some very tricky semantics!
- Extraction goal is a set of rules, which describes the semantic:

$$\texttt{SYMBOL} \rightarrow \texttt{IMPLICATION\_CONDITION}$$

- Slices of the model can be extracted!

Dependencies

- Dependencies are the easiest possibility way in Kconfig to express relations
- In Kconfig dependencies must always be fullfiled
- ```
  config HAVE_ARCH_ALLOC_REMAP
    depends on NUMA && X86_32
  ```
- When HAVE_ARCH_ALLOC_REMAP is enabled the dependencies must also be satisfied:
  $\texttt{HAVE\_ARCH\_ALLOC\_REMAP} \rightarrow (\texttt{NUMA} \wedge \texttt{X86\_32})$
- This only handles the boolean case.
  Tristate is more complicated, but very similar


  ```

Choices

- A choice gathers multiple other config options,
  were only one option is enabled
- Choices have no names, and can't be referenced, so we have to generate artificial keys.

```
choice "High Memory Support"
  config HIGHMEM4G  [...]  # These are 2
  config HIGHMEM64G [...]  # alternatives
endchoice
```

- $\texttt{CHOICE\_0} \rightarrow \texttt{HIGHMEM4G} \oplus \texttt{HIGHMEM64G}$
  $\texttt{HIGHMEM4G} \rightarrow \texttt{CHOICE\_0}$ $\texttt{HIGHMEM64G} \rightarrow \texttt{CHOICE\_0}$
- This only handles the boolean choices, tristates are *much* more complicated

Choices - optional

- A choice gathers multiple other config options, were only one or no option is enabled
- Choices have no names, and can't be referenced, so we have to generate artificial keys.

```
choice "High Memory Support" ; optional
  config HIGHMEM4G   [...]   # These are 2
  config HIGHMEM64G  [...]   # alternatives
endchoice
```

- $\texttt{CHOICE\_0} \rightarrow \texttt{HIGHMEM4G} \oplus \texttt{HIGHMEM64G} \oplus \texttt{\_\_FREE\_\_}$

  $\texttt{HIGHMEM4G} \rightarrow \texttt{CHOICE\_0} \qquad \texttt{HIGHMEM64G} \rightarrow \texttt{CHOICE\_0}$
- This only handles the boolean choices, tristates are *much* more complicated

selects - Reverse Dependencies

- A selection means, that enabling an option can cause an other option to be enabled automatically
- In Kconfig selects don't obey dependencies, but this behaviour is deprecated, rarely used and will hopefully be removed in the future
- A select can be guarded with an expression, then the select is only done, when the expression is true

```
config X86
  select HAVE_ARCH_GDB                  # X86 selects HAVE_ARCH_GDB
  select GENERIC_PENDING_IRC if SMP # only if SMP enabled
```

- $\texttt{X86} \rightarrow \texttt{HAVE\_ARCH\_GDB}$
- $\texttt{X86} \rightarrow (\texttt{SMP} \rightarrow \texttt{GENERIC\_PENDING\_IRQ})$

selects - revisited

- `selects` do not only have a forward reference
- If an option can't be visible in the menu, and has no defaule value, it can only be enabled by selects
- One of the selecting options must be true:
- ```
  config OLPC
    select GPIOLIB
  config ARCH_REQUIRE_GPIOLIB
    select GPIOLIB
  ```
- Here `GPIOLIB` is selected by two different options
- $\texttt{GPIOLIB} \rightarrow (\texttt{OLPC} \vee \texttt{ARCH\_REQUIRE\_GPIOLIB})$
- Default values (which can be guarded with expressions) can also be handled in a similar manner


  ```

The Kconfig model

config options	10924	dependencies	6919
boolean	5665	selects	3792
tristate	4482	choices	53
other	1777	within choices	186

Table: Used features in Kconfig: Linux 2.6.37-rc8, x86

- **11690** implication conditions,
  with no Kconfig specific construct left
- In average 5.4 options are mentioned on the right side

Conclusions

- `undertaker` is conceptionally not Linux-specific
- Exploits domain knowledge,
  but presence implications are project agnostic
- Variability models with implication conditions don't have to be complete, they can be under approximated

- Published as Free Software at
  `http://vamos.informatik.uni-erlangen.de/trac/undertaker`
- Feel free to get in touch with us!

Conclusions

- `undertaker` is conceptionally not Linux-specific
- Exploits domain knowledge,
  but presence implications are project agnostic
- Variability models with implication conditions don't have to be complete, they can be under approximated

- Published as Free Software at
  `http://vamos.informatik.uni-erlangen.de/trac/undertaker`
- Feel free to get in touch with us!

Thank you for your attention!