Disjoint Union Types in P0

Project 9 / Group 8

Jason Balaci

McMaster University

April 2021

Table of Contents

Table of Contents

What are Disjoint Union Types?

This is a text in the first frame. This is a text in the first frame. This is a text in the first frame.

How were they implemented?

Grammar Changes

- Text visible on slide 1

Grammar Changes

- Text visible on slide 1
- Text visible on slide 2

Grammar Changes

- Text visible on slide 1
- Text visible on slide 2
- Text visible on slide 3

Grammar Changes

- Text visible on slide 1
- Text visible on slide 2
- Text visible on slide 3
- Text visible on slide 4

Table of Contents

Example: Maybe & Either

Example: Lists

Example: Strings

Remark

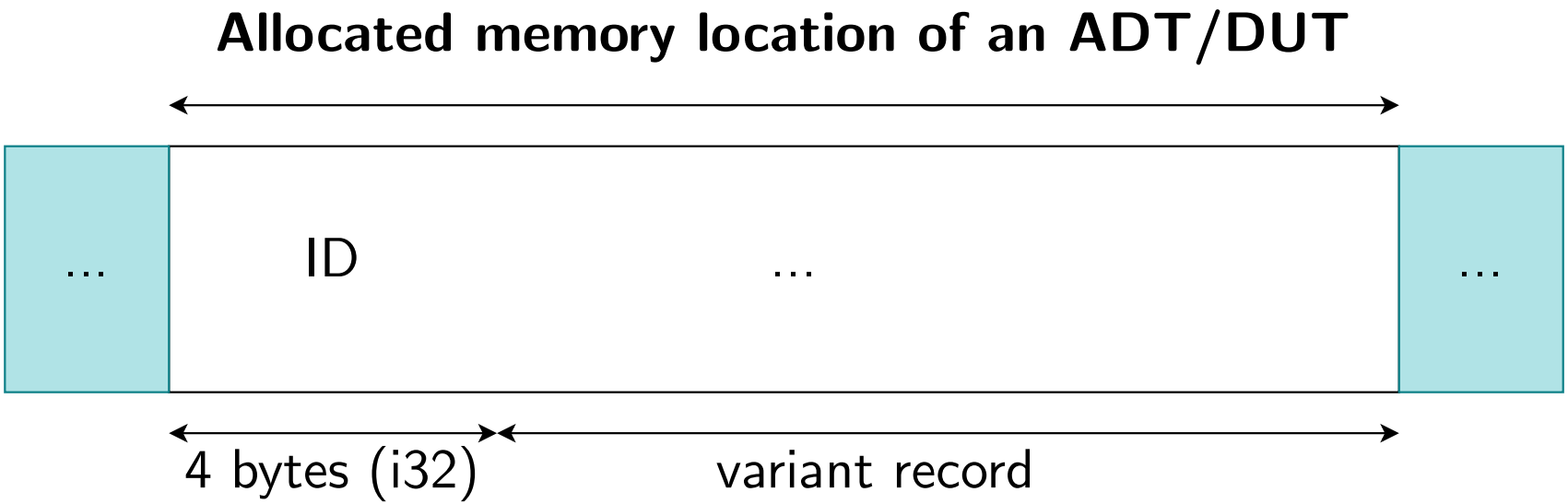
Sample text

Table of Contents

Memory Impact & Management

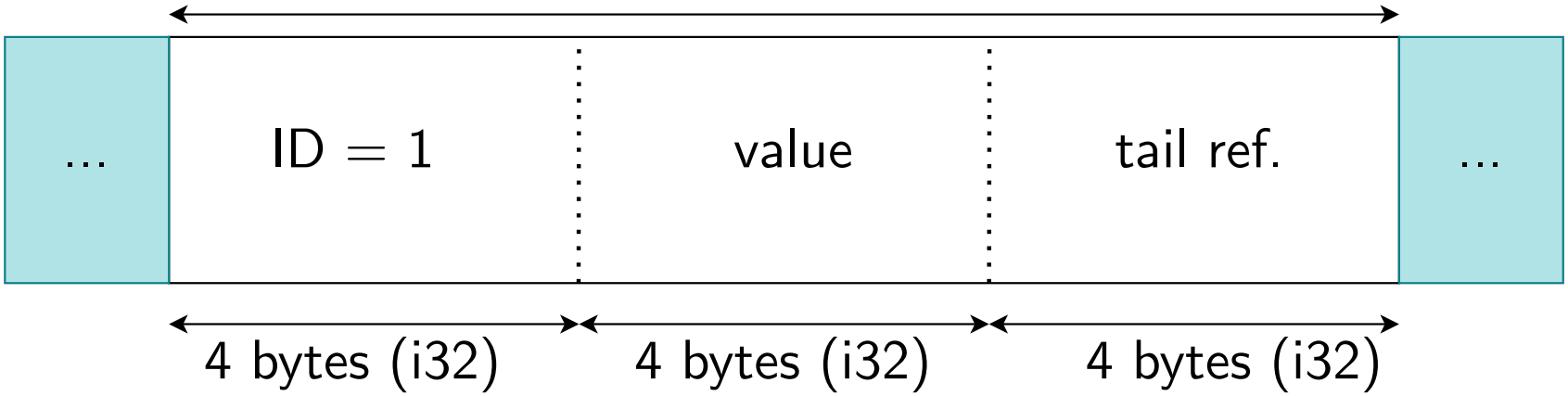
Each instance of a DUT is located on the heap, and instances of local/global DUTs are pointers to the locations of their corresponding DUT on the heap.

- Size of an allocation depends on the size of the variant being instantiated
- Offsets to accessing variables work similar to records, with a 4 byte offset for the variant id.

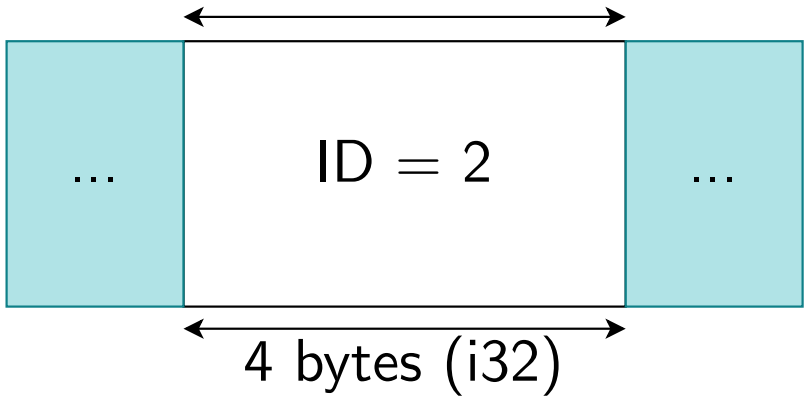


Example: Lists in Memory

Allocated memory location of a ‘Cons‘ (12 bytes)



Allocated memory location of a ‘Nil‘ (4 bytes)



Notable Design Decisions

- Text visible on slide 1

Notable Design Decisions

- Text visible on slide 1
- Text visible on slide 2

Notable Design Decisions

- Text visible on slide 1
- Text visible on slide 2
- Text visible on slide 3

Notable Design Decisions

- Text visible on slide 1
- Text visible on slide 2
- Text visible on slide 3
- Text visible on slide 4

Notes on Runtimes

- When working with DUTs/ADTs, we often tend to create recursive algorithms...

Notes on Runtimes

- When working with DUTs/ADTs, we often tend to create recursive algorithms...
- This causes issues for **pywasm**
  - Due to being interpreted in Python, a recursive call stack size limitation is imposed onto our programs.

Notes on Runtimes

- When working with DUTs/ADTs, we often tend to create recursive algorithms...
- This causes issues for **pywasm**
  - Due to being interpreted in Python, a recursive call stack size limitation is imposed onto our programs.
- Thankfully, **wasmer** has no issues!

Notes on Runtimes

- When working with DUTs/ADTs, we often tend to create recursive algorithms...
- This causes issues for **pywasm**
  - Due to being interpreted in Python, a recursive call stack size limitation is imposed onto our programs.
- Thankfully, **wasmer** has no issues!
- In-browser WebAssembly execution also has no issues, but we don't ship a web browser with the compiler.

Table of Contents

- Type variables!
  - Polymorphic disjoint union types! No more StringLists, IntLists, BooleanLists!
  - More code reuse!

- Type variables!
  - Polymorphic disjoint union types! No more StringLists, IntLists, BooleanLists!
  - More code reuse!
- More built-in types and syntactic sugars
  - Strings, Lists, Maps as a basic set of built-in DUTs
  - Stronger syntactic sugar for String generation (e.g., “abcd...” for quickly instantiating large strings)

- Type variables!
  - Polymorphic disjoint union types! No more StringLists, IntLists, BooleanLists!
  - More code reuse!
- More built-in types and syntactic sugars
  - Strings, Lists, Maps as a basic set of built-in DUTs
  - Stronger syntactic sugar for String generation (e.g., “abcd...” for quickly instantiating large strings)
- Improved Memory Management
  - Memory freeing!
  - Memory reuse!
  - Allocation specialization for built-in DUTs!

References