Haskell at Runtime

Zachary Stigall

Boulder Haskell Programmers

February 5, 2014

What Does GHC Do?

What Does GHC Do?

- Desugars Haskell code into Core.

What Does GHC Do?

- Desugars Haskell code into Core.

Definition

Core is GHC's intermediate language based on System FC, with a couple additions.

What Does GHC Do?

- Desugars Haskell code into Core.

Definition

Core is GHC's intermediate language based on System FC, with a couple additions.

Definition

System F is a Polymorphic Lambda Calculus language, FC adds GADTS to it.

What Does GHC Do?

- Desugars Haskell code into Core.
- Applies Optimizations to Core

Definition

Core is GHC's intermediate language based on System FC, with a couple additions.

Definition

System F is a Polymorphic Lambda Calculus language, FC adds GADTS to it.

What Does GHC Do?

- Desugars Haskell code into Core.
- Applies Optimizations to Core
- Converts Core into STG

Definition

Core is GHC's intermediate language based on System FC, with a couple additions.

Definition

System F is a Polymorphic Lambda Calculus language, FC adds GADTS to it.

What Does GHC Do?

- Desugars Haskell code into Core.
- Applies Optimizations to Core
- Converts Core into STG
- Compiles STG to C-- (kinda)

Definition

Core is GHC's intermediate language based on System FC, with a couple additions.

Definition

System F is a Polymorphic Lambda Calculus language, FC adds GADTS to it.

What Does GHC Do?

- Desugars Haskell code into Core.
- Applies Optimizations to Core
- Converts Core into STG
- Compiles STG to C-- (kinda)
- C-- is then compiled to LLVM / C / Machine Code

Definition

Core is GHC's intermediate language based on System FC, with a couple additions.

Definition

System F is a Polymorphic Lambda Calculus language, FC adds GADTS to it.

What Does GHC Do?

- Desugars Haskell code into Core.
- Applies Optimizations to Core
- Converts Core into STG
- Compiles STG to C-- (kinda)
- C-- is then compiled to LLVM / C / Machine Code
- Packs on the Haskell Runtime System (RTS)

Definition

Core is GHC's intermediate language based on System FC, with a couple additions.

Definition

System F is a Polymorphic Lambda Calculus language, FC adds GADTS to it.

RTS

The RTS provides:

- Garbage Collection

RTS

The RTS provides:

- Garbage Collection
- Scheduler

RTS

The RTS provides:

- Garbage Collection
- Scheduler
- Execution of non-compiled code

RTS

The RTS provides:

- Garbage Collection
- Scheduler
- Execution of non-compiled code
- Dynamic Linker

RTS

The RTS provides:

- Garbage Collection
- Scheduler
- Execution of non-compiled code
- Dynamic Linker
- Profiler

RTS

The RTS provides:

- Garbage Collection
- Scheduler
- Execution of non-compiled code
- Dynamic Linker
- Profiler
- Software Transactional Memory

GC

- Very Efficient - Generational (Out of necessity)

GC

- Very Efficient - Generational (Out of necessity)
- Data does not point to younger generations

GC

- Very Efficient - Generational (Out of necessity)
- Data does not point to younger generations
- This lets the GC just remove all non-pointed to data at GC

GC

- Very Efficient - Generational (Out of necessity)
- Data does not point to younger generations
- This lets the GC just remove all non-pointed to data at GC

Links:
Broad Overview
Way too in depth (ghc.haskell.org)

GC - Flags

These require program is compiled with '-rtsopts'.

GC - Flags

These require program is compiled with '-rtsopts'.
This does not cover all options, just the most common.

GC - Flags

These require program is compiled with '-rtsopts'.
This does not cover all options, just the most common.

- -A[size] sets allocated area for GC (default 512K)

GC - Flags

These require program is compiled with '-rtsopts'.
This does not cover all options, just the most common.

- -A[size] sets allocated area for GC (default 512K)
- -H[size] suggested Heap size. (size optional)

GC - Flags

These require program is compiled with '-rtsopts'.
This does not cover all options, just the most common.

- -A[size] sets allocated area for GC (default 512K)
- -H[size] suggested Heap size. (size optional)
- -c[n] does compacting, use only when you need to drastically reduce RAM usage

GC - Flags

These require program is compiled with '-rtsopts'.
This does not cover all options, just the most common.

- -A[size] sets allocated area for GC (default 512K)
- -H[size] suggested Heap size. (size optional)
- -c[n] does compacting, use only when you need to drastically reduce RAM usage
- -qg parallel GC

GC - Flags

These require program is compiled with '-rtsopts'.
This does not cover all options, just the most common.

- -A[size] sets allocated area for GC (default 512K)
- -H[size] suggested Heap size. (size optional)
- -c[n] does compacting, use only when you need to drastically reduce RAM usage
- -qg parallel GC

Recommended ghc-gc-tune - Allows profiling of varying -A and -H parameters

Scheduler

Horribly Complicated

- Centered around the Run Queue

Scheduler

Horribly Complicated

- Centered around the Run Queue
- The Run Queue is a dispatcher for Capabilities

Scheduler

Horribly Complicated

- Centered around the Run Queue
- The Run Queue is a dispatcher for Capabilities
- Capabilities are virtual CPU's that execute Thread State Objects (TSO)

Scheduler

Horribly Complicated

- Centered around the Run Queue
- The Run Queue is a dispatcher for Capabilities
- Capabilities are virtual CPU's that execute Thread State Objects (TSO)
- TSO's are essentially closures and are GC'ed

Scheduler

Horribly Complicated

- Centered around the Run Queue
- The Run Queue is a dispatcher for Capabilities
- Capabilities are virtual CPU's that execute Thread State Objects (TSO)
- TSO's are essentially closures and are GC'ed

Links:
Very Detailed (ghc.haskell.org)
Nice blog post by Edward Z. Yang

Basic Profiling

- All profiling will require the program compiled with -rtsopts

Basic Profiling

- All profiling will require the program compiled with -rtsopts
- For some basic (but very detailed) profiling compile with -prof -fprof-auto -rtsopts

Basic Profiling

- All profiling will require the program compiled with -rtsopts
- For some basic (but very detailed) profiling compile with -prof -fprof-auto -rtsopts
- Run with Prog [ARGS] +RTS [RTS-OPTS]

Basic Profiling

- All profiling will require the program compiled with -rtsopts
- For some basic (but very detailed) profiling compile with -prof -fprof-auto -rtsopts
- Run with Prog [ARGS] +RTS [RTS-OPTS]
- -s gives a summary

Basic Profiling

- All profiling will require the program compiled with -rtsopts
- For some basic (but very detailed) profiling compile with -prof -fprof-auto -rtsopts
- Run with Prog [ARGS] +RTS [RTS-OPTS]
- -s gives a summary
- -p writes Prog.prof with tons of details

Basic Profiling

- All profiling will require the program compiled with -rtsopts
- For some basic (but very detailed) profiling compile with -prof -fprof-auto -rtsopts
- Run with Prog [ARGS] +RTS [RTS-OPTS]
- -s gives a summary
- -p writes Prog.prof with tons of details

Demo

Basic Profiling

- All profiling will require the program compiled with -rtsopts
- For some basic (but very detailed) profiling compile with -prof -fprof-auto -rtsopts
- Run with Prog [ARGS] +RTS [RTS-OPTS]
- -s gives a summary
- -p writes Prog.prof with tons of details

Demo

- You can also tell GHC that you want analysis on a specific part by inserting your own cost center

Basic Profiling

- All profiling will require the program compiled with -rtsopts
- For some basic (but very detailed) profiling compile with -prof -fprof-auto -rtsopts
- Run with Prog [ARGS] +RTS [RTS-OPTS]
- -s gives a summary
- -p writes Prog.prof with tons of details

Demo

- You can also tell GHC that you want analysis on a specific part by inserting your own cost center
- {-# SCC "CC-Name" #-} (expression)

Basic Profiling

- All profiling will require the program compiled with -rtsopts
- For some basic (but very detailed) profiling compile with -prof -fprof-auto -rtsopts
- Run with Prog [ARGS] +RTS [RTS-OPTS]
- -s gives a summary
- -p writes Prog.prof with tons of details

Demo

- You can also tell GHC that you want analysis on a specific part by inserting your own cost center
- {-# SCC "CC-Name" #-} (expression)

More Demo

Heap Profiling

Same compile opts as last time

Heap Profiling

Same compile opts as last time

-hy as an RTS-OPT, produces a Prog.hp, this can be converted to postscript with hp2ps

Heap Profiling

Same compile opts as last time

-hy as an RTS-OPT, produces a Prog.hp, this can be converted to postscript with hp2ps

Demo

Other Profiling

- -B Sound Bell at GC

Other Profiling

- -B Sound Bell at GC
- -xc Print Stack Trace on exception

Other Profiling

- -B Sound Bell at GC
- -xc Print Stack Trace on exception
- -M Set Max Heap size

That's It

Questions?
My GitHub: `http://github.com/ZirroStig`