



# Merrimac Compiler Effort

---

Jayanth Gummaraju

Mattan Erez

Manman Ren

Alan Wray

Aug 12, 2003

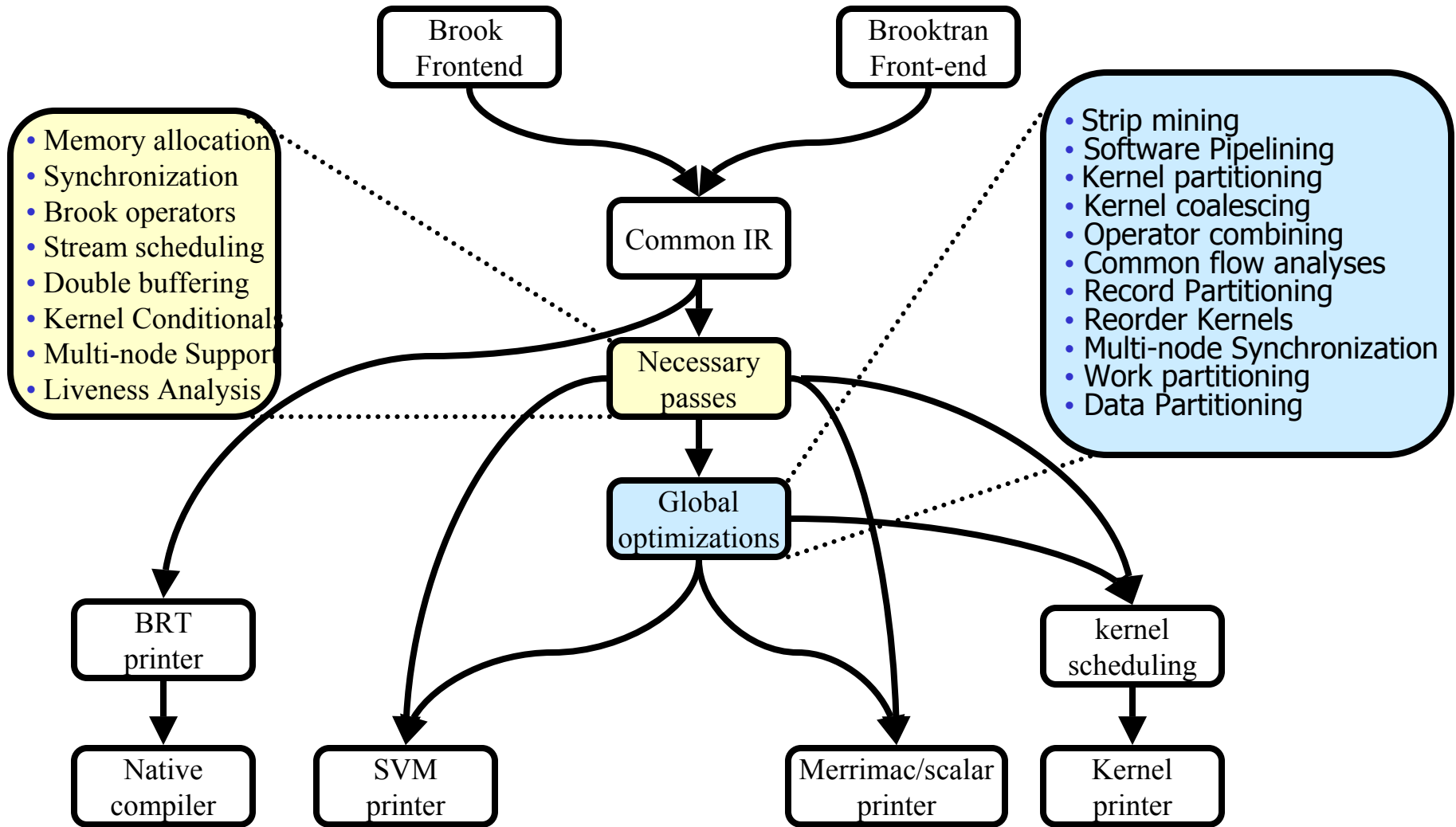


# Outline

---

- Compiler Goals
- Open64 Status
- Reservoir compiler Vs Open64
- “Leveraging” Reservoir compiler

# Compiler Goals





# Open64 Implementation Plan

---

- Implement brook and brooktran front ends
  - Modify gcc and fortran front ends of Open64
  - Generate High WHIRL representation
- Implement necessary/optimization passes in back end on the High WHIRL representation
  - Generates an optimized High WHIRL representation
- Implement BRT, SVM, and Merrimac Printers
  - Modify the tool “whirl2c” which works on High WHIRL
- Develop interfaces with kernel scheduler/low level compiler



# Open64 Status: Brook Front End

---

- Parsing: Add keywords to lex tables
  - Variable qualifiers: stream, memstream, out, reduce
  - Function qualifier : kernel
  - UPC qualifiers : relaxed, shared, strict
- Propagate keywords into gcc tree
  - Set flags in gcc tree for the qualifiers
- Propagate keywords into WHIRL tables
  - Set flags in WHIRL symbol tables
  
- Main issues
  - Files absent in gcc front end distribution
  - No “space” in WHIRL type qualifier table



# Interface to Back End (1)

## Queries on streams

Bool TY_is_stream (TY_IDX ty_idx)	Is this a stream?
BOOL TY_is_memstream (TY_IDX ty_idx)	Is this a memstream?
BOOL TY_is_outstream (TY_IDX ty_idx)	Is this an “out” stream?
BOOL TY_is_reduce (TY_IDX ty_idx)	Is this scalar being “reduced”?
BOOL TY_is_grouped_stream (TY_IDX ty_idx)	Is this a grouped stream?
BOOL TY_is_derived_stream (TY_IDX ty_idx)	Is this a stream derived from stencil/group/domain?
BOOL TY_is_stencil_stream (TY_IDX ty_idx)	Is this a stencil stream?

## Queries on functions

BOOL PU_is_kernel_function (const PU& pu)	Is this a kernel function?
BOOL Intr_is_stream_operator (const INTRINSIC I)	Is this intrinsic function a brook stream operator?



# Interface to Back End (2)

## Stream properties in overloaded Array Table

ST_IDX ARB_derived_var (const ARB_HANDLE stream)	Symbol Table index for stream from which the stencil/group stream is derived
INT64 ARB_stride_val (const ARB_HANDLE stream)	Stride value for a strided derived stream
BOOL ARB_const_shape (const ARB_HANDLE stream)	Does the stream have a constant shape?
BOOL ARB_dynamic_shape (const ARB_HANDLE stream)	Does the stream have a dynamically changing shape?
STR_BND ARB_lbnd_boundary (const ARB_HANDLE stream)	Stencil boundary condition for left side (in that dimension)
STR_BND ARB_ubnd_boundary (const ARB_HANDLE stream)	Stencil boundary condition for right side (in that dimension)
INT64 ARB_lbnd_val (const ARB_HANDLE stream)	Value of lower bound of stencil/group/domain if constant
INT64 ARB_ubnd_val (const ARB_HANDLE stream)	Value of upper bound of stencil/group/domain if constant

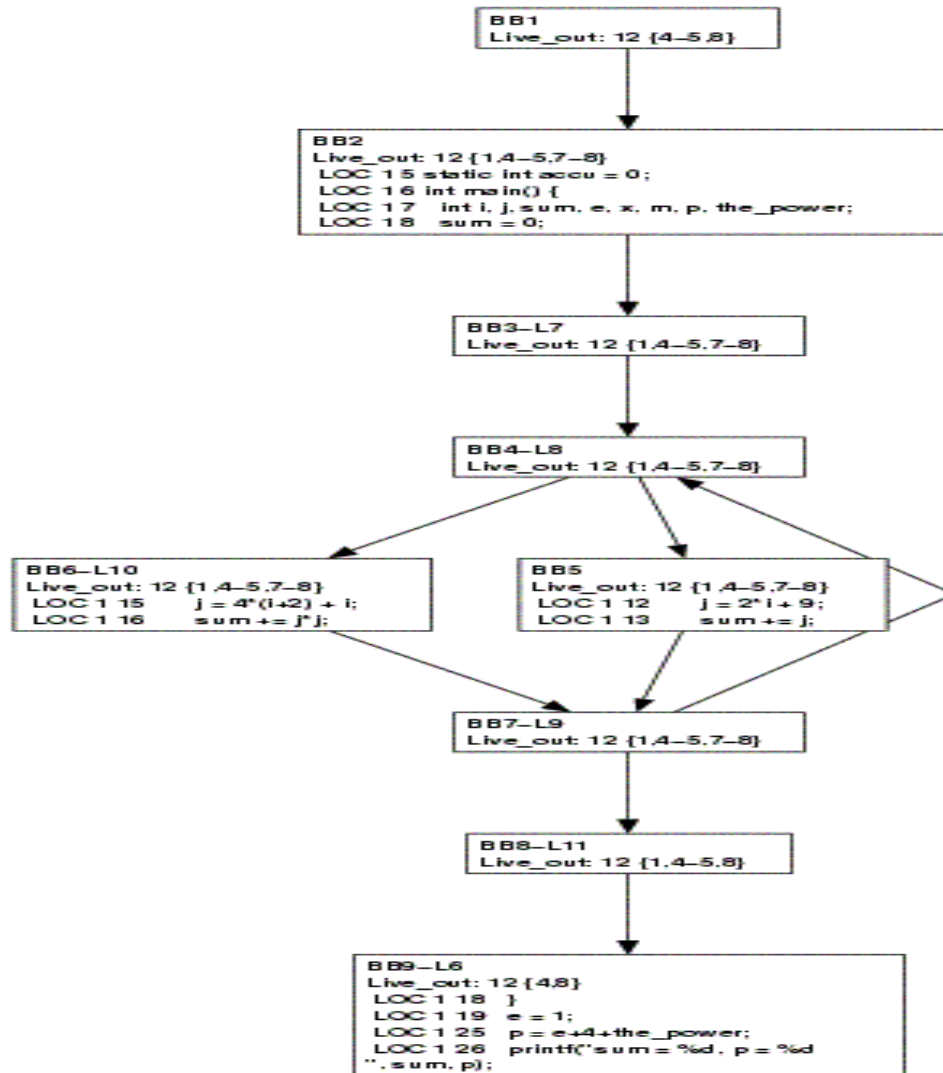


# Necessary Passes

---

- Liveness Analysis implementation
  - Live variables after/before every statement (implemented)
- Brook Operators
  - Treated as intrinsic function calls
  - Get handle into intrinsic table in front end
  - “Expand” the operators in back end (done in Ino phase)
- Memory allocation/deallocation of streams, scalar stream synchronization
  - Designed algorithms to implement these passes

# Liveness Analysis Example





# Ongoing work

---

- Final stages of Front end implementation
  - Some more flags to be set
  - Propagate qualifiers when “typedef” is used
- Brook Runtime Printer
  - Alan is working on whirl2brt
- Error checks
  - Check declarations
  - Match declarations of actual and formal kernel parameters
- Test cases
  - Write test cases to verify sanctity of the compiler

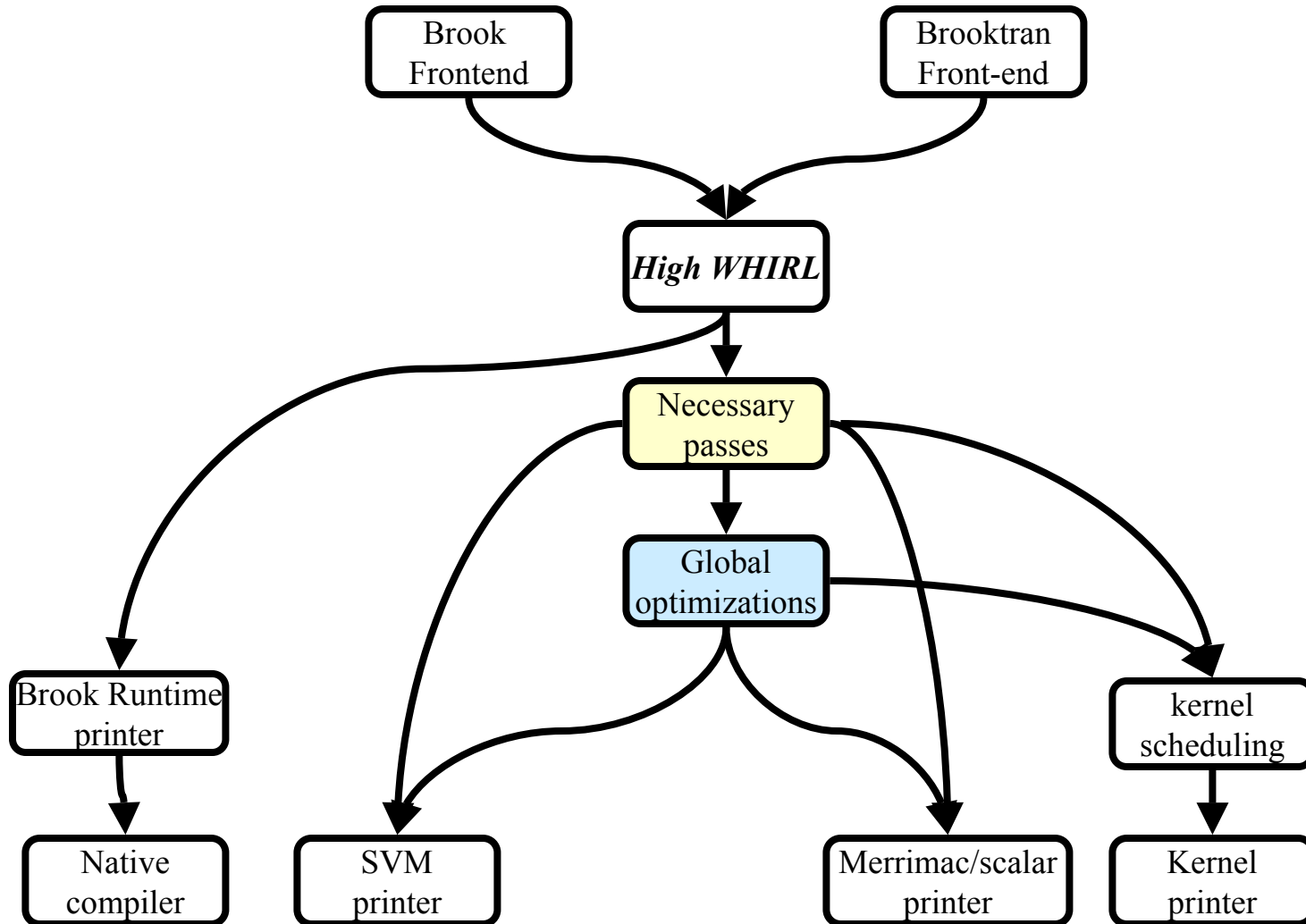


# Open64 Vs Reservoir Compiler

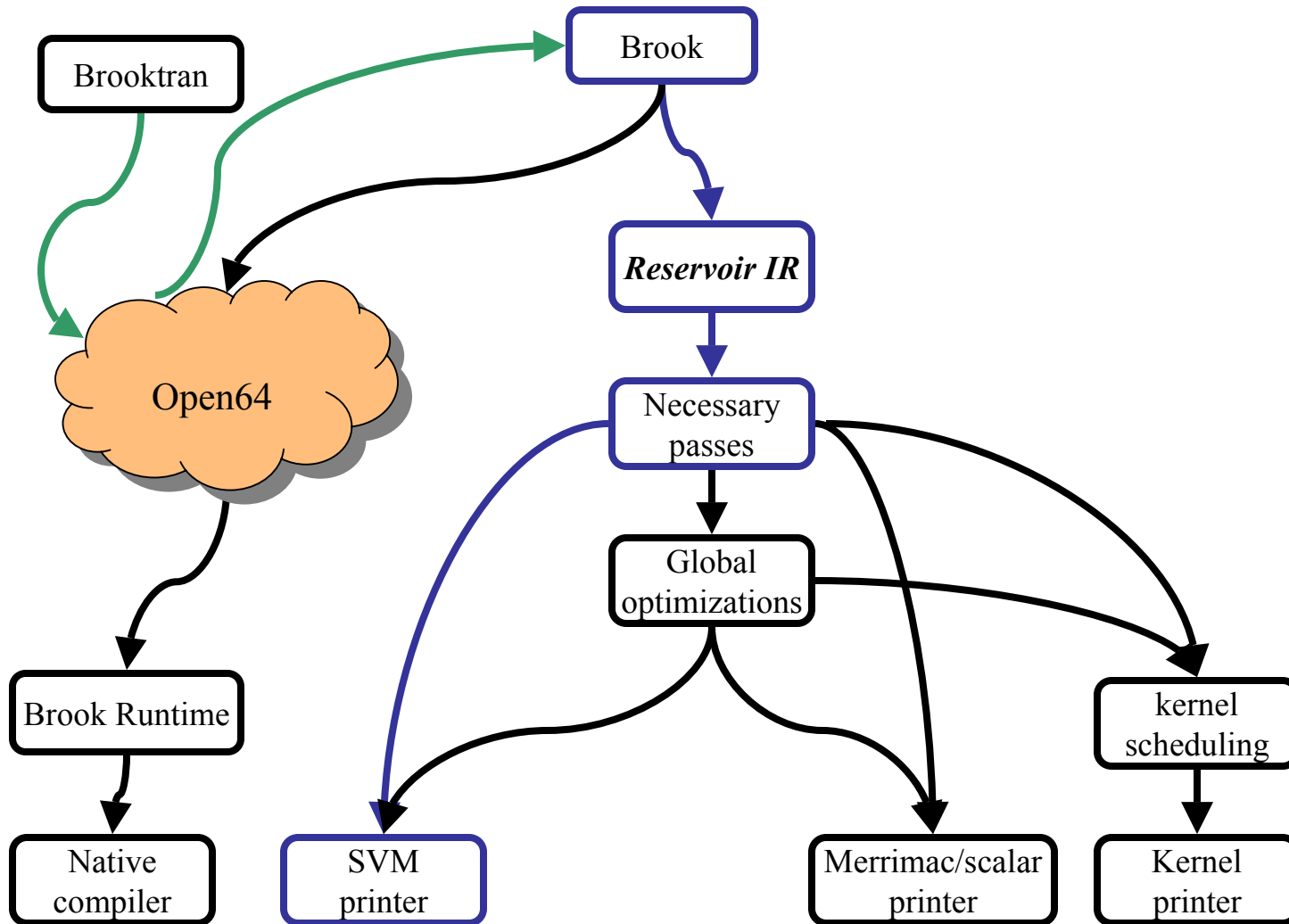
---

- Open64
  - Gone through cold start period
  - Have complete control
  - Freely distributable
  - Fairly robust – used in several places
  - Has Fortran/C++ front end
- Reservoir Compiler
  - Effort reuse
  - Several people working on it
  - More time
  - More experience

# Compilation Flowchart (recap)



# Leveraging Reservoir Compiler





# Retargeting Reservoir Compiler to Merrimac

---

- Merrimac printer
- UPC multi-node support
- ScatterOp/GatherOp support
  - Reservoir may not support it
- Conditionals inside kernels
  - Predication Vs Conditional Streams
- Multidimensional stream optimizations
- Stream Scheduling (specific to Merrimac)
- Other Global Optimizations



# Questions

---

- UPC?
  - What does Reservoir use for multi-node memory model?
- Compiler Optimizations/Passes
  - What optimizations does reservoir plan to implement?
- When will we get access to the Reservoir compiler?
- Open64 or Reservoir compiler?