A Modified Approach to Data Cache Management

Gary Tyson
Matthew Farrens
John Matthews
Andrew Pleszkun

Motivation

- Most cache misses are caused by a few load instructions
- Current cache designs have simple allocation policies
  - Allocate on cache miss (Read Miss)
- Improved allocation can improve performance

Proposal

- ✓ **Target cache allocation to only those loads that demonstrate locality**
- ✓ **Cacheable/Non-Allocatable (CNA)**
  - Static analysis
  - Dynamic analysis
- ✓ **Optimize for bandwidth**

Outline of Presentation

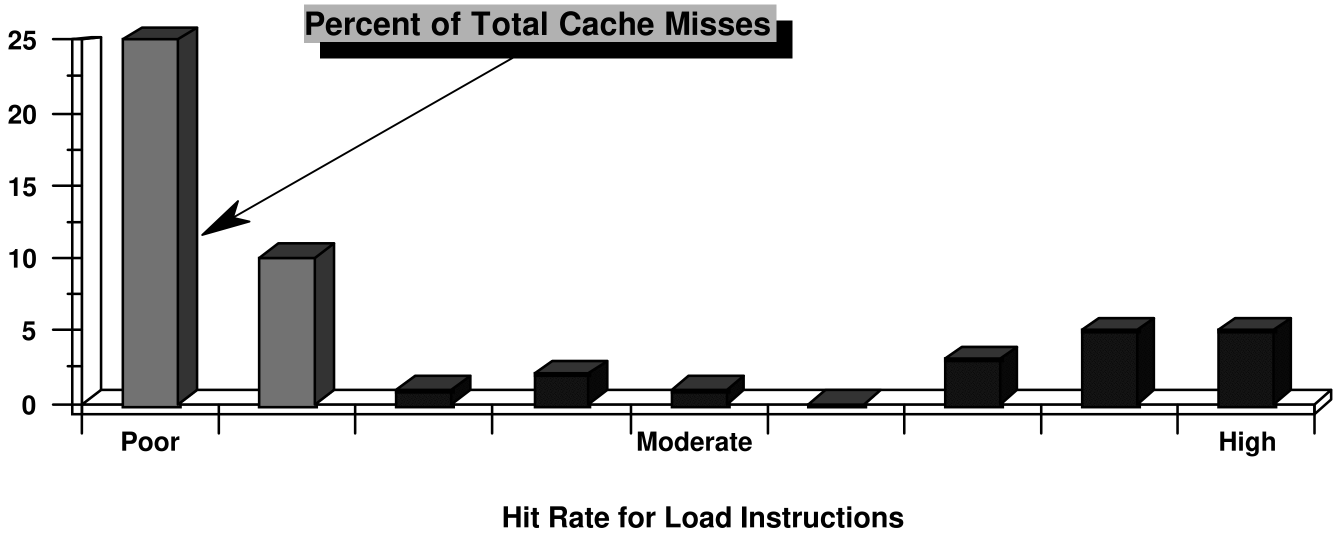
- ✓ Previous work
- ✓ Cache behavior
- ✓ Modified allocation methods
- ✓ Conclusions and future work

Previous Related Work

- ✓ **Software prefetch schemes**
  - Abraham, et. al. [1993]
- ✓ **Hardware prefetch schemes**
  - Chen and Bear [1995]
- ✓ **Streaming buffers**
  - Jouppi [1990]

Cache Hit Rates

- Target instructions with poor cache hit rate (shown in red).



Allocation Strategy

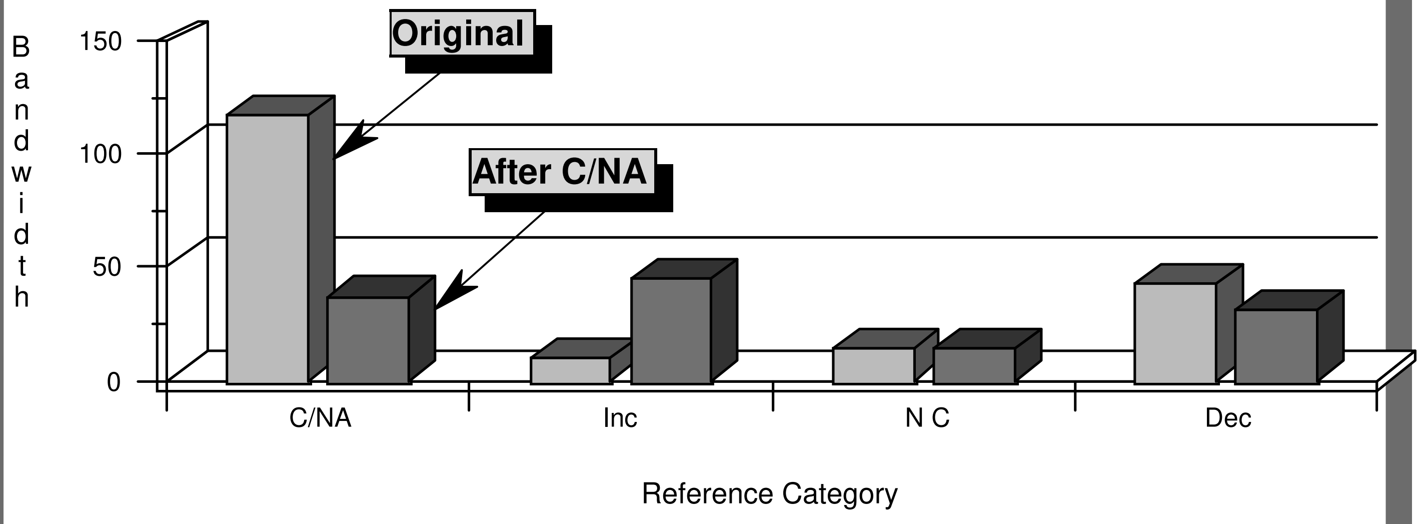
- **How many instructions must be targeted?**
  - Very few (approx. 1% of load instructions)
- **How can these instructions be identified?**
  - Based on previous cache hit behavior

Static Cache Allocation Policy

- ✓ **Compiler Selection of Cache Allocation Policy**
  - Requires instruction set support
  - Requires profile information
- ✓ **Determine which load instructions have poor cache performance**
  - Don’t Allocate if past hit rate is poor

Effect on Bandwidth

- ✓ 30% decrease in bandwidth
- ✓ Significant increase in cache misses

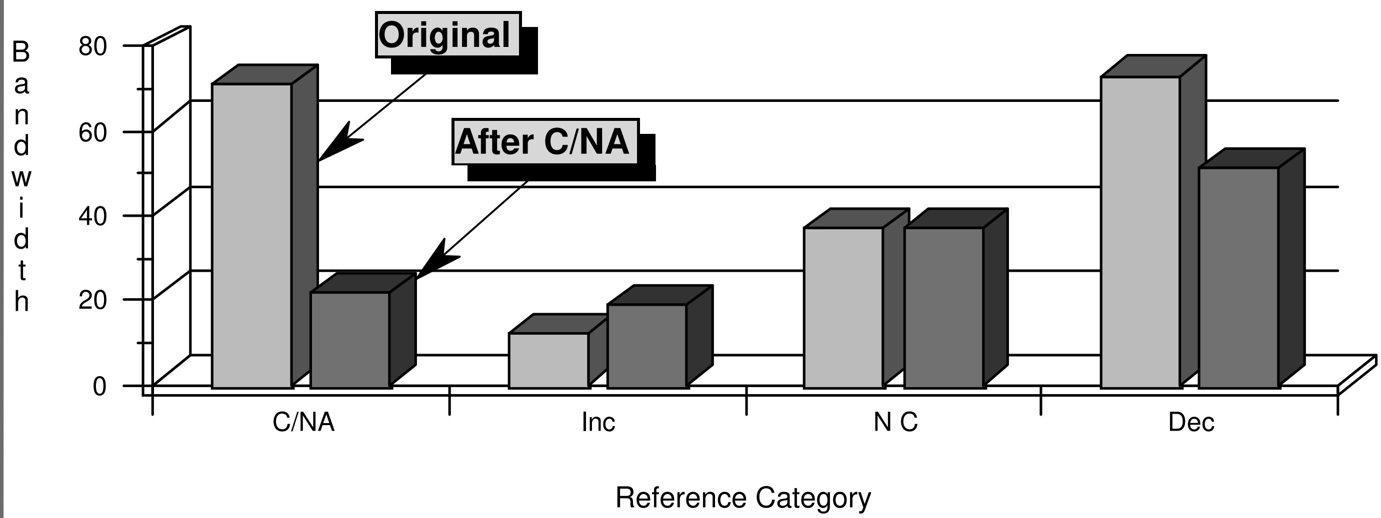


Improving Static Analysis

- **Reduce number of C/NA loads**
  - Some loads with poor hit rate perform a useful allocation for other instructions
- **Modify static analysis to include the usefulness of the load**

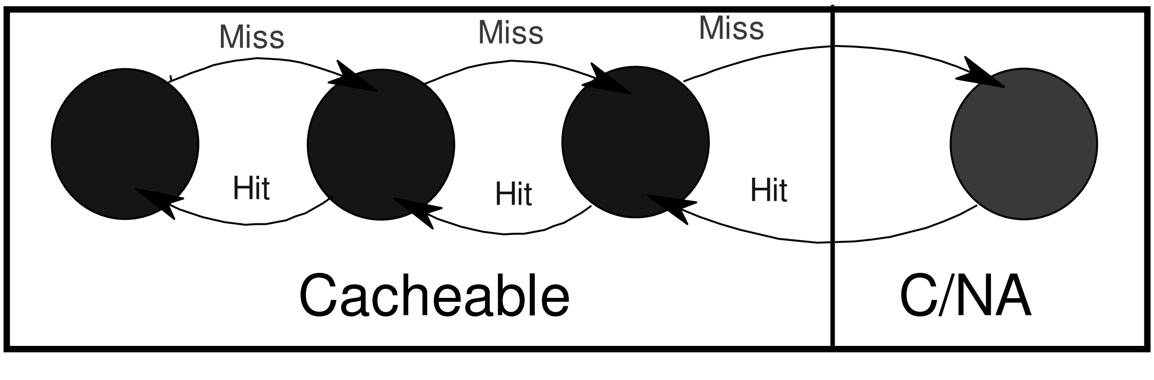
Effect on Bandwidth

- ✓ 30% decrease in bandwidth
- ✓ Almost no change in hit rate



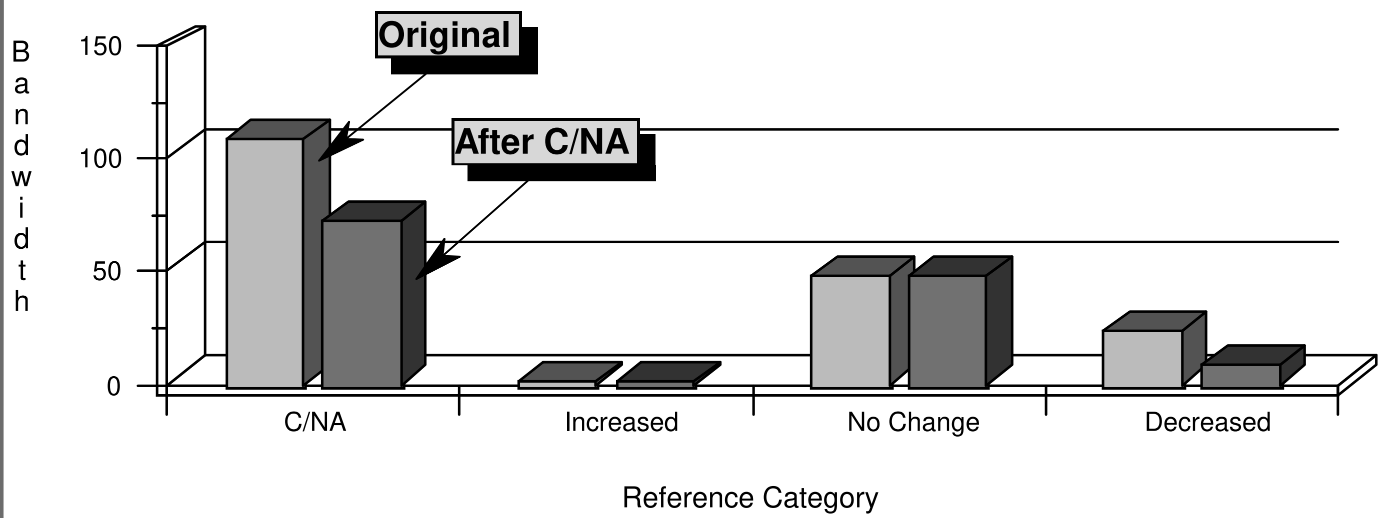
Dynamic Approach to C/NA

- Use branch prediction techniques
- 2-bit history table with 3 cacheable states and 1 C/NA state



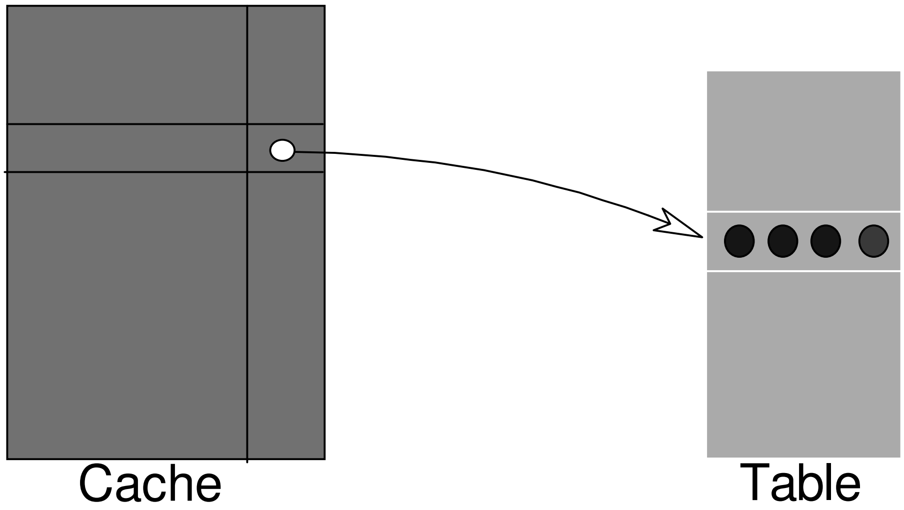
Effect on Bandwidth

✓ Similar characteristics to static



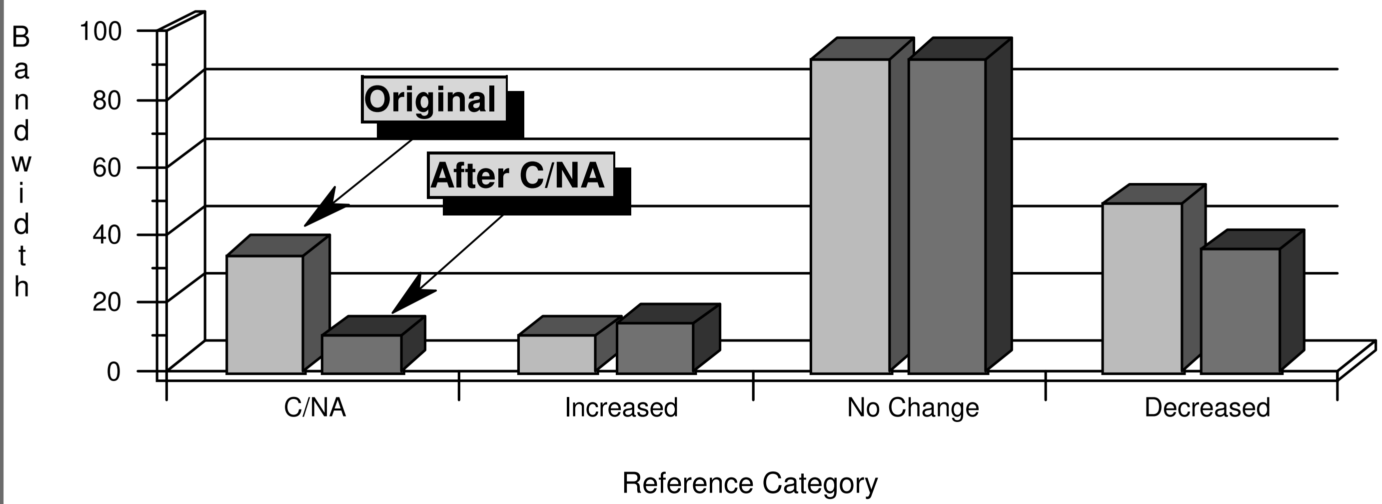
Improving the Dynamic Scheme

- Allow cache hits on a line to decrement the 2-bit counter



Effect on Bandwidth

- ✓ 20% reduction in bandwidth
- ✓ No ISA change or profiling required



Conclusions

- ✓ Branch prediction techniques can (and should) be applied to memory operations
- ✓ Selective allocation can improve memory system performance
- ✓ These techniques can be applied to a large set of problems

Future Work

- ✓ **Allocation for multi-level caches**
- ✓ **Data movement in cache consistency schemes for shared memory systems**
  - dynamically switch between write update and write invalidate schemes
- ✓ **Selective cache prefetching**
- ✓ **Target cache hit rate**

Optimizing for Hit Rate

- **Can we optimize for hit rate?**
  - Yes (20% improvement for 8K direct mapped cache)
- **Why?**
  - Reduced contention for cache lines
  - Transient addresses don't effect working set

Questions?

Gary Tyson

tyson@cs.ucr.edu

http://cs.ucr.edu/~tyson/tech/8-95

A Modified Approach to Data Cache Management

Gary Tyson
Matthew Farrens
John Matthews
Andrew Pleszkun

Motivation

- Most cache misses are caused by a few load instructions
- Current cache designs have simple allocation policies
  - Allocate on cache miss (Read Miss)
- Improved allocation can improve performance

Proposal

- ✓ **Target cache allocation to only those loads that demonstrate locality**
- ✓ **Cacheable/Non-Allocatable (CNA)**
  - Static analysis
  - Dynamic analysis
- ✓ **Optimize for bandwidth**

Outline of Presentation

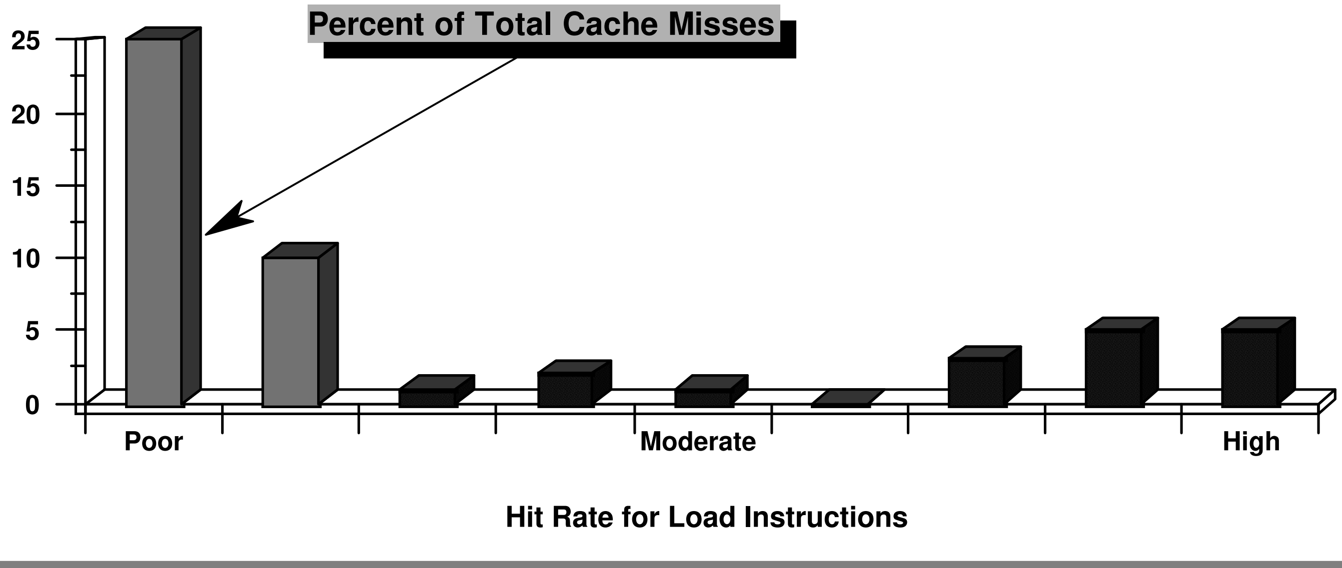
- ✓ Previous work
- ✓ Cache behavior
- ✓ Modified allocation methods
- ✓ Conclusions and future work

Previous Related Work

- ✓ **Software prefetch schemes**
  - Abraham, et. al. [1993]
- ✓ **Hardware prefetch schemes**
  - Chen and Bear [1995]
- ✓ **Streaming buffers**
  - Jouppi [1990]

Cache Hit Rates

- ✓ **Target instructions with poor cache hit rate (shown in red).**



Allocation Strategy

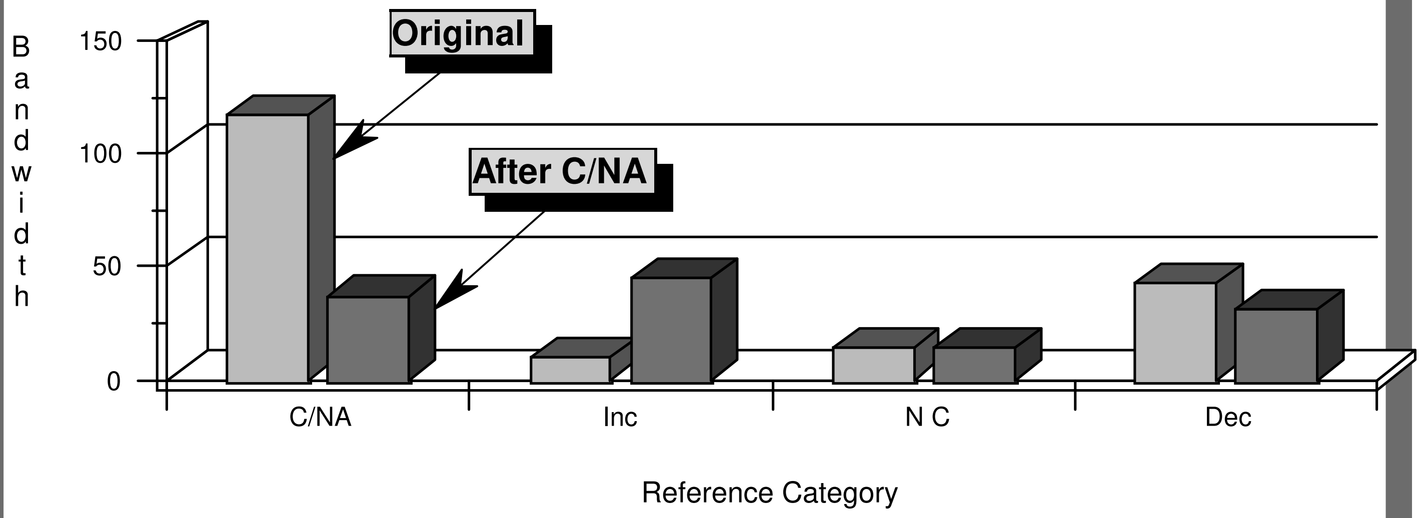
- **How many instructions must be targeted?**
  - Very few (approx. 1% of load instructions)
- **How can these instructions be identified?**
  - Based on previous cache hit behavior

Static Cache Allocation Policy

- **Compiler Selection of Cache Allocation Policy**
  - Requires instruction set support
  - Requires profile information
- **Determine which load instructions have poor cache performance**
  - Don’t Allocate if past hit rate is poor

Effect on Bandwidth

- ✓ 30% decrease in bandwidth
- ✓ Significant increase in cache misses

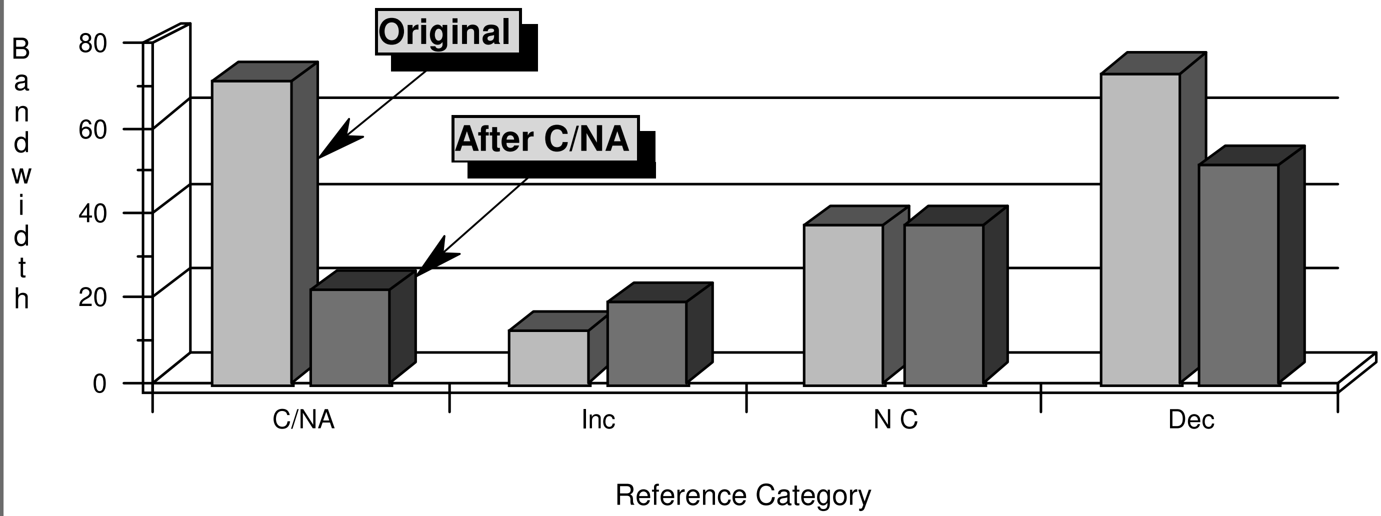


Improving Static Analysis

- ✓ **Reduce number of C/NA loads**
  - Some loads with poor hit rate perform a useful allocation for other instructions
- ✓ **Modify static analysis to include the usefulness of the load**

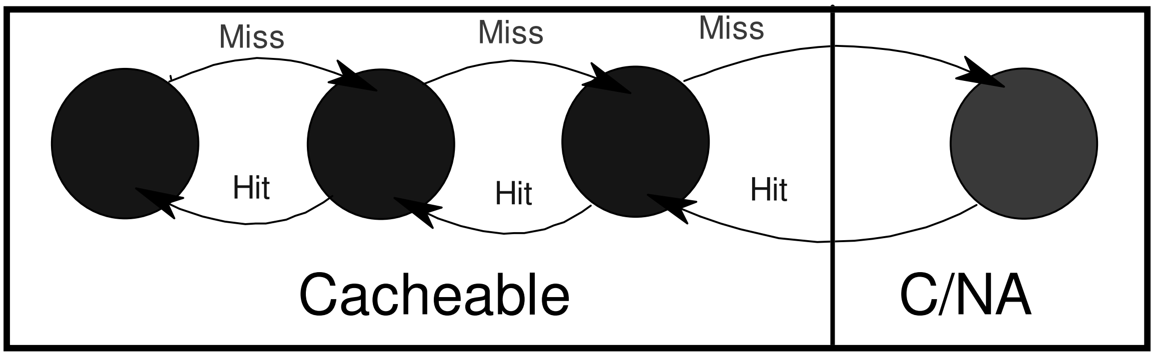
Effect on Bandwidth

- ✓ 30% decrease in bandwidth
- ✓ Almost no change in hit rate



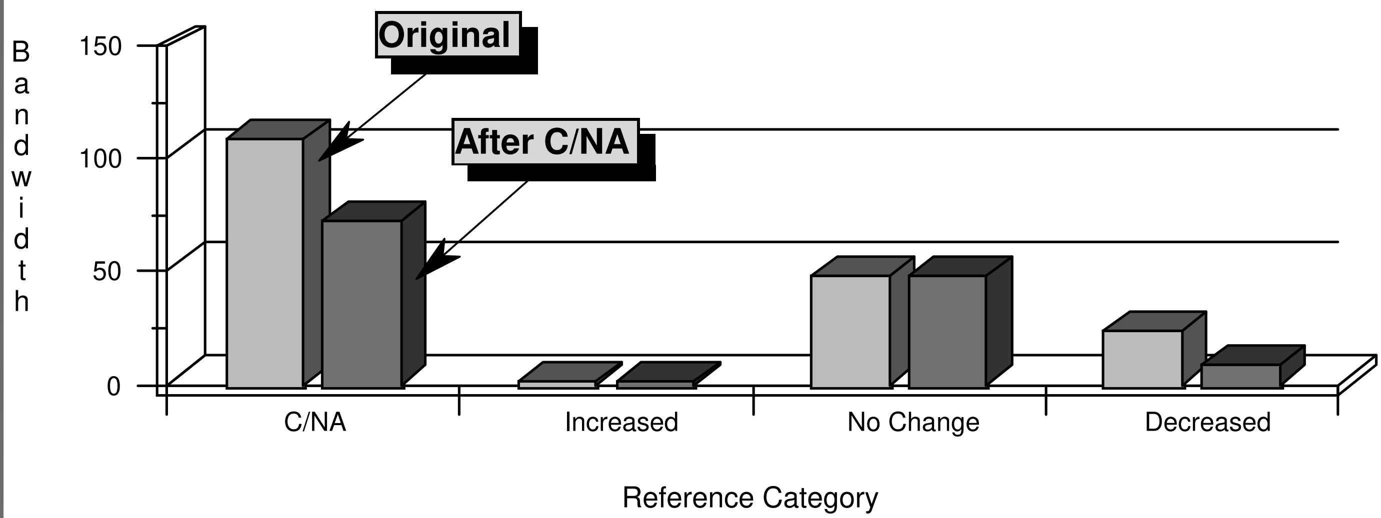
Dynamic Approach to C/NA

- ✓ **Use branch prediction techniques**
- ✓ **2-bit history table with 3 cacheable states and 1 C/NA state**



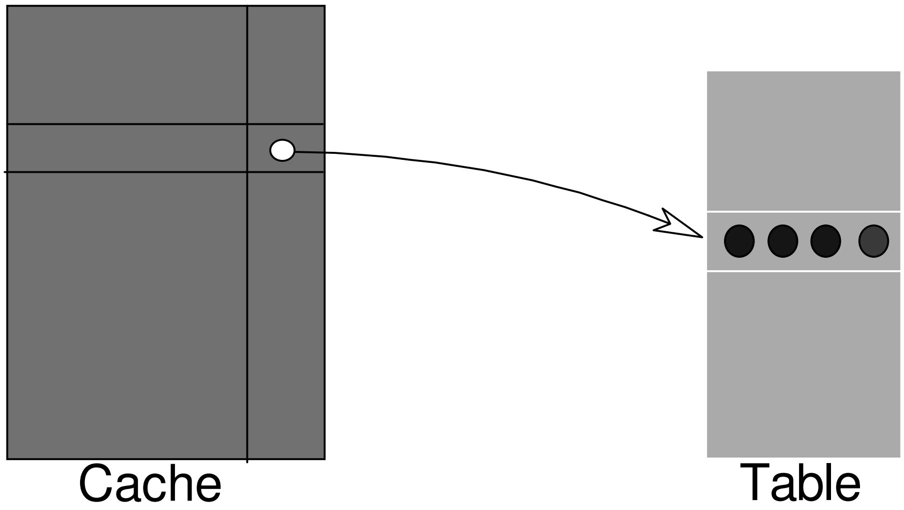
Effect on Bandwidth

✓ **Similar characteristics to static**



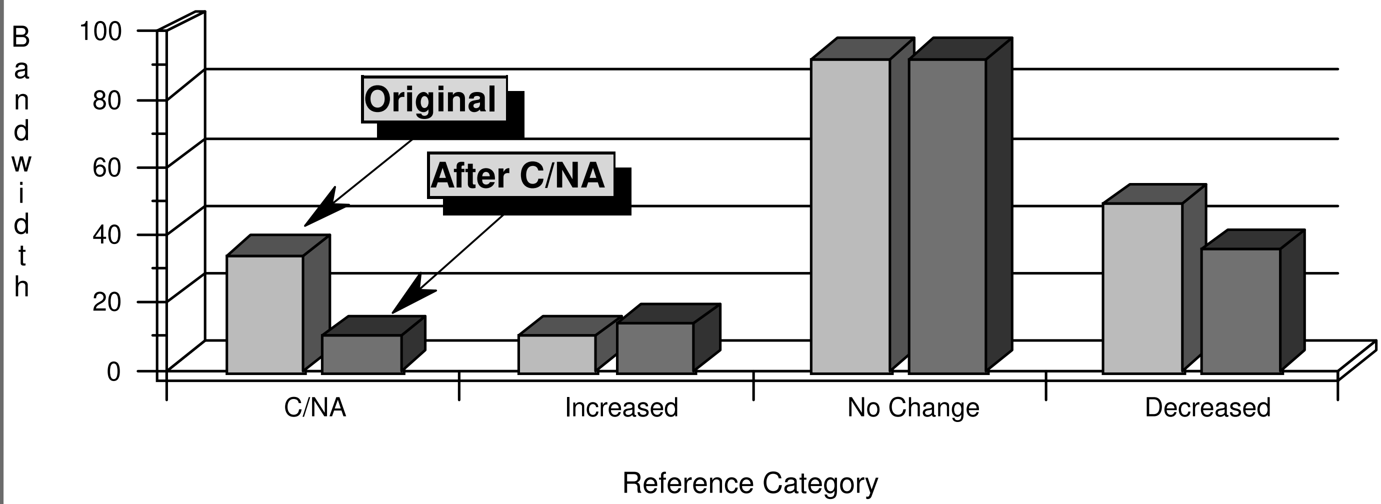
Improving the Dynamic Scheme

- Allow cache hits on a line to decrement the 2-bit counter



Effect on Bandwidth

- ✓ 20% reduction in bandwidth
- ✓ No ISA change or profiling required



Conclusions

- ✓ Branch prediction techniques can (and should) be applied to memory operations
- ✓ Selective allocation can improve memory system performance
- ✓ These techniques can be applied to a large set of problems

Future Work

- ✓ **Allocation for multi-level caches**
- ✓ **Data movement in cache consistency schemes for shared memory systems**
  - dynamically switch between write update and write invalidate schemes
- ✓ **Selective cache prefetching**
- ✓ **Target cache hit rate**

Optimizing for Hit Rate

- ✓ **Can we optimize for hit rate?**
  - Yes (20% improvement for 8K direct mapped cache)
- ✓ **Why?**
  - Reduced contention for cache lines
  - Transient addresses don't effect working set

Questions?

Gary Tyson

tyson@cs.ucr.edu

http://cs.ucr.edu/~tyson/tech/8-95