Distributed Automated Deduction:

an Introduction to the Clause-Diffusion methodology

Maria Paola Bonacina

(joint work with Jieh Hsiang)

Outline

- Introduction and motivation:
  - why parallel deduction
  - why contraction-based strategies

- Analysis of the problems in the parallelization of deduction strategies.

- Overview of the Clause-Diffusion methodology for distributed deduction.

Why parallel deduction

Strategy : $\mathcal{C} = < I ; \Sigma >$

I : inference system

Σ : search plan

Derivation :

$$S_0 \vdash_{\mathcal{C}} S_1 \vdash_{\mathcal{C}} \ldots \vdash_{\mathcal{C}} S_i \vdash_{\mathcal{C}} \ldots$$

$$\left.\begin{array}{l} \bullet \text{ Refutational completeness of } I \\ \bullet \text{ Fairness of } \Sigma \end{array}\right\} \text{completeness of } \mathcal{C}$$

- $\boxed{\text{Efficiency}}$ of $\mathcal{C}$

Contraction-based strategies

- Expansion rules : $\dfrac{S}{S'}$ $S \subset S'$

  e.g. resolution,
  paramodulation.

- Contraction rules : $\dfrac{S}{S'}$ $S \not\subset S'$

  e.g. (conditional) simplification,
  normalization,
  subsumption.

- At the operational level:

  forward contraction,

  backward contraction.

Contraction-based strategies

Use eagerly contraction rules to contain the growth of the generated search space;

may feature:

- orderings on terms and clauses,
- contraction-first search plans,
- ordering-based restrictions to expansion.

Why contraction-based strategies

- Some of the most successful theorem provers are contraction-based: SbReve 2, SBR3, Otter, RRL, Reveal ....

- Known challenge:
  prove completeness in the presence of contraction rules.

- A new challenge:
  parallel contraction-based strategies.

Analysis of the problems in the parallelization of deduction strategies

Classification of strategies for the purpose of parallelization

- Subgoal-reduction strategies.

- Expansion-oriented strategies.

- Contraction-based strategies.

Subgoal-reduction strategies

Form of the derivation:

$$(S\,;\,\varphi_0\,;\,A_0) \vdash_{\mathcal{C}} (S\,;\,\varphi_1\,;\,A_1) \vdash_{\mathcal{C}} \ldots \vdash_{\mathcal{C}} (S\,;\,\varphi_i\,;\,A_i) \vdash_{\mathcal{C}} \ldots$$

Examples:

- functional programming,
- term rewriting,
- logic programming,
- PTTP.

Property: $\boxed{\text{static data base.}}$

Expansion-oriented strategies

Form of the derivation:

$$(S_0; N_0) \vdash_e (S_1; N_1) \vdash_e \dots \vdash_e (S_i; N_i) \vdash_e \dots$$

Expansion:

$$\frac{S \cup \{\psi_1, \psi_2\};\ N}{S \cup \{\psi_1, \psi_2\};\ N \cup \{\psi_3\}}$$

"raw clause"

Possibly forward contraction:

$$\psi_3 \xrightarrow{*} \psi_3'$$

ψ_3' ? delete / add to S

No backward contraction

Property: monotonically increasing data base

$$S_0 \subseteq S_1 \subseteq \dots \subseteq S_i \subseteq S_{i+1} \subseteq \dots$$

Contraction-based strategies

Form of the derivation:

$$S_0 \vdash_{\mathcal{E}} S_1 \vdash_{\mathcal{E}} \dots \vdash_{\mathcal{E}} S_i \vdash_{\mathcal{E}} S_{i+1} \vdash_{\mathcal{E}} \dots$$

Expansion: $\dfrac{S}{S'}$ $S \subset S'$

Both forward and backward contraction: $\dfrac{S}{S'}$ $S \not\subseteq S'$

Property : $\boxed{\text{highly dynamic data base}}$

$\forall i$ $S_i \subseteq S_{i+1}$ or $S_i \not\subseteq S_{i+1}$

Granularity of parallelism

	granularity of data	granularity of operations
parallelism at the term level (fine grain)	TERM	SUBTASK OF INFERENCE STEP
parallelism at the clause level (medium grain)	CLAUSE	INFERENCE STEP

coarse grain parallelism

Conflicts

Concurrent processes access the same grain of data:

- write-write conflicts
  between contraction steps,

- read-write conflicts
  between contraction steps,

- read-write conflicts
  between expansion steps and
  contraction steps.

Read-write conflicts are caused
by backward contraction.

Parallelization of subgoal-reduction strategies

- Static data base:
  - + Pre-processing
  - + Read-only data
  - + Specialized data structures

- No conflicts

A grain of data can be as small as a term (fine-grain).

Parallelization of expansion-oriented strategies

- Monotonically increasing data base:
  - + No pre-processing of all clauses at compile-time,
  - + S_i as a whole not read-only (single clauses are).
- Conflicts:
  - + Read-write conflicts: NO.
  - + Write-write conflicts: YES.
- Change of scale.

⇩

Term level granularity is too fine.

Parallelization of contraction-based strategies

- Highly dynamic data base:
  - + No pre-processing of all clauses at compile-time,
  - + No read-only data and higher rate of write-accesses (against shared memory),
  - + All data play both roles of "simplifier" and "simplified" (against specialized data structures).

- Conflicts:
  - + Read-write conflicts : YES
  - + Write-write conflicts : YES

The backward contraction bottleneck

- Fine / medium grain
- Shared memory

One backward contraction step induces many.

Concurrent backward contraction processes : conflicts.

Just one backward contraction process : bottleneck.

Types of parallelism and strategies

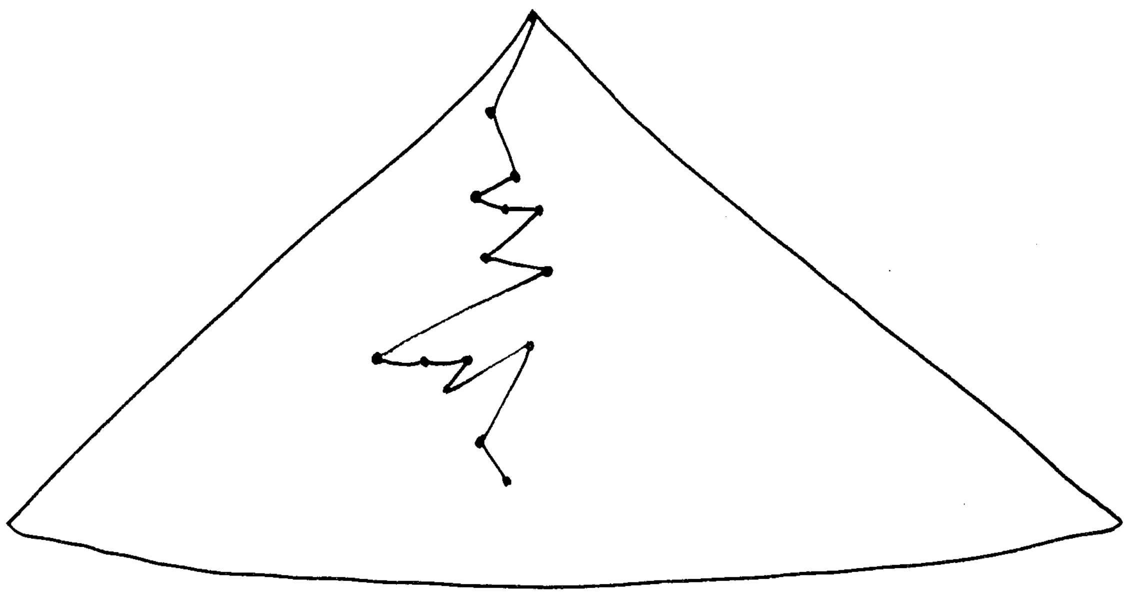
	subgoal-reduction	expansion-oriented	contraction-based
parallelism at the term level	✓		
parallelism at the clause level	✓	✓	
coarse grain	✓	✓	✓

What kind of coarse grain parallelism for automated deduction ?

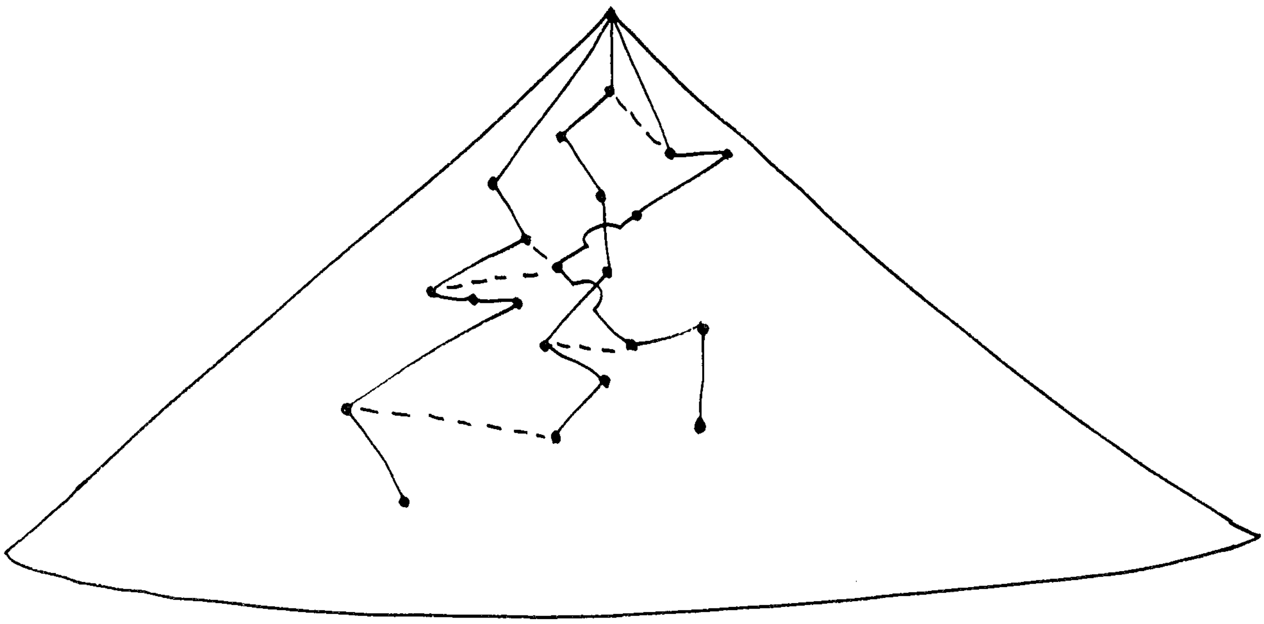
Parallelism at the search level

Parallelism at the search level

Sequential search:



Parallel search:



—— : inference steps

- - - : communication steps

Parallelism at the search level

Concurrent, asynchronous, loosely-coupled deductive processes

develop their own derivations

by working on separate sets of clauses (no conflicts)

and

by exchanging clauses as messages.

Success is reached as soon as one of the processes succeeds.

Distributed environment

- Purely distributed:
  - asynchronous, loosely coupled processors (nodes),
  - distributed memory,
  - communication by message passing.
- Mixed shared - distributed:
  - also a shared memory component
  - combines message passing with communication through memory.

+ Networks of computers
+ Asynchronous multi-processors with distributed memory

Overview of the Clause - Diffusion methodology

Partition the search space

At the clause level :

subdivide the data base of clauses.

For all clauses φ, assign φ to a process p_i :

$$\underbrace{S^0 \cup \ldots \cup S^i \cup \ldots \cup S^{n-1}}_{} = S$$

"residents" at p_i (S^i)

global data base (S)

Communication of clauses

Each process p_i takes care of the inferences on S^i, but it is not guaranteed to find a proof by using S^i only.

⇓

Each process sends its residents to the others in form of "inference messages"

$$(S^i \, ; \, M^i)$$

residents

received inference messages

Partition the search space

At the inference level:

expansion inferences:

if $\psi_1 \in S^i$ and $\psi_2 \in \mathcal{M}^i$

P_i paramodulates ψ_2 into ψ_1 but not vice versa.

It also prevents the systematic duplication of expansion steps.

It applies also to other rules, e.g. resolution, hyper-resolution and unit-resulting resolution.

No general subdivision of contraction inferences based on ownership of clauses.

In a contraction-based strategy, each process tries to contract as much as possible residents and messages before expansion and communication.

Local contraction (w.r.t. S^i)

and global contraction

w.r.t. $\bigcup_i S^i$) by schemes for

distributed global contraction.

A Clause-Diffusion strategy

is specified by

the set of inference rules,

the search plan which schedules
contraction steps
expansion steps
communication steps
at each process,

the algorithm to allocate clauses
("new settlers") as residents,

the mechanism for message-passing,

the scheme for distributed
global contraction.

Distributed derivation

$$(S_i, M_i, CP_i, NS)_0^k \vdash_{e}$$

$$(S_i, M_i, CP_i, NS)_1^k \vdash_{e} \ldots$$

$$\ldots \vdash_{e} (S_i, M_i, CP_i, NS)_i^k \vdash_{e} \ldots$$

$1 \leq k \leq n$

S : residents,

M : inference messages,

CP : raw clauses,

NS : new settlers.

Not covered in this talk

- Schemes for distributed global contraction:
  - distributed, communication-oriented
  - distributed, duplication-oriented
  - mixed shared-distributed
- Distributed allocation of clauses
- Message passing: interaction of contraction steps on messages and communication
- Schedule of types of operations at each process
- Distributed fairness
- Distributed subsumption
- Discarding redundant messages
- Implementations:
  - Aquarius
  - Peers

Summary

- Contraction - based strategies
- Analysis of classes of strategies and parallelism
- Backward contraction
- The backward contraction bottleneck
- Coarse grain parallelism and distributed memory
- Parallelism at the search level: no synchronization / redundancy.
- Overview of the Clause - Diffusion methodology
- Distributed derivations

Current and future work

- Study of parallel search
  - partioning a search space:
    blind / informed
  - parallel search plans
- Fine-tuning of implementations:
  - heuristics for the allocation of clauses
  - reconstructing a distributed proof
  - experiments.