## ■地の漂流者たち

沢木耕太郎さんの,
自衛隊，アングラ演劇，ピンク映画，歌謡曲，沖縄の人,
集団就職 ( 川崎の人 ) の取材によるルポ．若者に焦点が当てられている．
先が見えない恐怖,
大きな力に身が委ねられている ( そして気づいていない ) 恐怖，そんな話.
特に年代を意識せずに読んでいたら 70 年代の社会の話だった.
... あれ今とたいして変わらなくないか？
言い訳って恰好わるいな，とか，少くとも元気でいよう，とか,
そんなことを考えたりした.
ちなみに，電車の中で読んでいたため，ふと目にはいった
「総裁選，揺れる国会」とか「現代・性のすべて」なんていう
吊り広告が，いつもより生々しく感じられた.

## ■ Automatic Data movement and Computation Mapping for Multi-level Parallel Architectures with Explicitly Managed Memories

スクラッチパッドメモリのデータを自動的に効率良くマネジメント
複数のレベルでの並列性による一般のプログラムからの効率的な割り当て

- creation buffers in on-chip memories
  - for holding portions of data accessed in a computation block
- automatic determination of array access functions
- generation of code
  - that moves data between slow off-chip memory and fast local memories

3. Automatic Data Management in Scratchpad Memories

plyhedral model をベースにした議論

- automatic allocation of storage space in scratchpad memories for holding portions of data accessed in a block of a program
- determination of access functions of references to arrays in scratchpad memories
- automatic generation of code for moving data between scratchpad memory and off-chip memory.

3.1 Details of the Framework

入力 = 配列のアクセス関数のようなプロウラムブロックの中の文の iteration spaces.
配列を A，与えられたプログラムの各文 S_{k} としたとき，read reference functions の行列を F_{k}^{l}，write reference function の行列を G_{k} とすると，データスペースは，読み書きそれぞれ,

```
DS_{r}^{A} = F_{k}^{l} I_{k}
DS_{w}^{A} = G_{k} I_{k}
```

で，全体のデータスペース DS_{rw}^{A} はその和集合.

で，問題は，これをオーバラップなく最大の disjoint sets に分割.
これは無向グラフ中の connected components を探す問題に等しい.
無向グラフのノードは，DS_{rw}^{A} の各データ.
ノード間にエッジがあるのは，ノード同士の共通集合が空でないとき.

3.1.1 Determining Local Memory Storage

- When the rank of the access matrix of an array reference is les than the iteration space dimensionality of the statement in which it is accessed, the data elements of the array accessed in the reference are said to have an order of magnitude (ore non-constant) reuse.
  - the partition is marked as beneficial to be copied to scratchpad memory
- Constatn reuse in the set is estimated by consdering each pair of data spaces, determining the volume of their intersection, and summing up thse volumes
  - determined by a fraction \delta <= emprirically fixed a value of 30% for \delta

3.1.2 Determining Access Functions of Local memory Array References

CLooG を使う

3.1.3 Generating Data Movement Code

Figure 1 に例題が示されている.

- We generate the loop structure of the code that moves data from global memory to local memory by scanning the selected data spaces using CLooG.
- CLooG scans the data spaces in an efficient way such that the generated loop structure leads to single load/store of each data element that is read/written even if the accessed data spaces of references are overlapping.

3.1.4 Optimizing Data movement

- The optimal strategy for determining data elements that need to be copied in and copied out requires data dependence information.
- In future work we plan to implement the optimization outlined above, based on data dependence information.

4 Tiling for Multiple Levels of Parallelism

4.1 Details of the Approach

- アーキテクチャ
  - a slow global memory
  - a set of parallel units at an outer level that communicate with each other thorough the global memory space
  - a set of parallel units within each outer-level parallel unit
  - a local fast explicitly managed scratchpad memory within each outer-level parallel unit shared by the inner-level parallel units

並列性の抽出には Boundhugula らのフレームワークを利用

## ■文献リスト

## ■作戦会議

楽しかった.