Static Analysis of Array Contents

Mathias Péron, Nicolas Halbwachs

Verimag/CNRS
Grenoble
Considering arrays in static analysis

- array bound checking: solved in 90% cases
- array contents:
  - expansion (small known size)
  - summarization
  - symbolic partitioning + summarization
Array summarization in Astrée

- Abstract each array $A$ by a single variable $a$
- Interpretation: $\psi(a) \iff \forall \ell = 1..n, \psi(A[\ell])$
- Each assignment $A[i] := \text{exp}$ is interpreted as a weak assignment to $a$:
  \[
  a \sqcup = \text{exp} \equiv a := \text{exp} [] a := a
  \]

Problems:

- no information gained from conditionals
- weak assignment can only lose information
- information about the initial content of arrays must be obtained by other means
Symbolic partitioning + summarization (1/3)

[Gopan, Reps, Sagiv - Popl05]

- partition each array into symbolic slices, e.g.,
  \[ A_1 = A[1..i-1], \ A_2 = A[i], \ A_3 = A[i+1..n] \]

- abstract each slice \( A_p \) by a single variable \( a_p \)

- interpret \( A[i] := \text{exp} \) as a strong assignment \( a_2 := \text{exp} \)

- interpret an index incrementation \( i++ \) as
  \[
  a_1 \sqcup = a_2 ; \ a_2 := a_3
  \]

Interpretation: \( \psi(a_p) \iff \forall \ell \in l_p, \psi(A[\ell]) \)
Symbolic partitioning + summarization (2/3)

m := A[1]
for(i:=2; i\leq n; i++){
  if (A[i]>m)
    m := A[i];
}

M. Péron, N. Halbwachs (Verimag/CNRS)
Symbolic partitioning + summarization (2/3)

Partition: $l_1 = [1..i - 1]$, $l_2 = \{i\}$, $l_3 = [i + 1..n]$

\[
m := A[1] \\
\text{for}(i:=2; i \leq n; i++)\{ \\
\quad \text{if } (A[i] > m) \\
\quad \quad m := A[i]; \\
\}\]
Symbolic partitioning + summarization (2/3)

Partition: \( l_1 = [1..i - 1], l_2 = \{i\}, l_3 = [i + 1..n] \)

\[
\begin{align*}
    m &:= A[1] \\
    \text{for}(i:=2; i\leq n; i++) \{ & \text{for } (i:=2; i\leq n; i++) \{ \\
        \text{if } (A[i] > m) & \text{if } (a_2 > m) \\
        m &:= A[i]; \\
        m &:= a_2; \\
        a_1 \sqcup = a_2; a_2 &:= a_3
    \}
\end{align*}
\]
Symbolic partitioning + summarization (2/3)

Partition: $l_1 = [1..i−1]$, $l_2 = \{i\}$, $l_3 = [i+1..n]$

$m := A[1]$

\[
\text{for}(i:=2; i \leq n; i++)\{
\text{if} \ (A[i] > m) \ m := A[i];
\}
\]

\[
\text{if} \ (a_2 > m) \ m := a_2; \quad 2 \leq i \leq n, m = a_2 \geq a_1
\]

\[
\text{a}_1 \sqcup = a_2; a_2 := a_3 \quad 2 \leq i \leq n, m \geq a_1, m \geq a_2
\]

\[
i = n+1, m \geq a_1
\]
Symbolic partitioning + summarization (3/3)

- able to discover unary properties about array elements
- unable to discover relations between array elements
- able to check (with TVLA) such relations (provided by the user)
  e.g., $\forall \ell = 1..n, A[\ell] = B[\ell]$
Symbolic partitioning + summarization (3/3)

- able to discover unary properties about array elements
- unable to discover relations between array elements
- able to check (with TVLA) such relations (provided by the user)
  
e.g., $\forall \ell = 1..n, A[\ell] = B[\ell]$

This work: generalization to discover simple relations:

$$\forall \ell \in I, \psi(A_1[\ell + k_1], \ldots, A_m[\ell + k_m])$$

for “simple programs” (pointwise relations between array slices, inspired by Lustre-V4)
Simple programs

- one-dimensional arrays
- simple traversal: \( i := e_1; \text{while}(\text{cond}) \{ \ldots ; i++ \} \)
- simple array access: \( A[i] := \exp(B[i+k]) \)
Simple programs

- one-dimensional arrays
- simple traversal: \(i:=e1; \text{while}(\text{cond})\{\ldots ; i++\}\)
- simple array access: \(A[i] := \text{exp}(B[i+k])\)

Examples:

for \(i:=1\) to \(n\) do
    \(A[i] := B[i]\);
end

\(m:= A[1];\)
for \(i:=2\) to \(n\) do
    if \(m < A[i]\) then \(m := A[i]\) endif
end

for \(i:=2\) to \(n\) do
    \(x:= A[i]; j:=i-1;\)
    while \(j \geq 1\) and \(A[j] > x\) do
        \(A[j+1]:=A[j]; j:=j-1\)
    end
    \(A[j+1]:= x\)
end

\(I\) set of index variables, \(I' = I \cup \{\ell\}\)
\(C\) set of content variables, \(A\) set of arrays.

M. Péron, N. Halbwachs (Verimag/CNRS)
Partitions and Properties (1/3)

Two basic lattices:

- $L_N (\ni \varphi)$: properties of indexes (at least DBMs)
- $L_C (\ni \psi)$: properties of contents

Partition: $\{ \varphi_p \}_{p \in P} \subset L_N (\mathcal{I}')$ such that

$$\bigvee_{p \in P} \varphi_p = \top_N \quad \text{and} \quad p \neq p' \Rightarrow \varphi_p \cap \varphi_{p'} = \bot_N$$

ex: $\varphi_1 = (1 \leq \ell < i)$, $\varphi_2 = (1 \leq \ell = i \leq n)$, $\varphi_3 = (i < \ell \leq n)$
Partitions and Properties (2/3)

Slice variables: \( \{ a_p^z \} \), \( a \mapsto A \in \mathcal{A}, p \in P, z \in \mathbb{Z} \)

\( a_p^z \) represents the slice \( A[\ell + z \mid \varphi_p(\ell)] \)

Properties: given a partition \( \{ \varphi_p \}_{p \in P} \),

\[ \Psi = (\varphi, (\psi_p)_{p \in P}) \]

\[ \gamma(\Psi) = \{ (\mathcal{I}, \mathcal{C}, \mathcal{A}) \text{ such that } \]
\[ \varphi(\mathcal{I}), \]
\[ \forall p \in P, \forall \ell, \quad \varphi_p(\mathcal{I} \cup \{\ell\}) \Rightarrow \psi_p[A[\ell + z]/a_p^z] \} \]
### Examples:

<table>
<thead>
<tr>
<th>( \varphi )</th>
<th>( \varphi_1 )</th>
<th>( \varphi_2 )</th>
<th>( \varphi_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( i = n + 1 )</td>
<td>((1 \leq \ell &lt; i))</td>
<td>((\ell = i \leq n))</td>
<td>((i &lt; \ell \leq n))</td>
</tr>
<tr>
<td>( \psi_1 )</td>
<td>(a_1^0 = b_1^0)</td>
<td>(\psi_2 = \text{False})</td>
<td>(\psi_3 = \text{False})</td>
</tr>
<tr>
<td>( \varphi )</td>
<td>((\ell = 1))</td>
<td>((2 \leq \ell &lt; i))</td>
<td>((\ell = i))</td>
</tr>
<tr>
<td>( \psi_1 )</td>
<td>(a_1^0 \leq a_1^1)</td>
<td>(a_2^0 \geq a_2^{-1})</td>
<td>(a_3^0 = x)</td>
</tr>
</tbody>
</table>
### Partitions and Properties (3/3)

**Examples:**

<table>
<thead>
<tr>
<th>( \varphi_1 )</th>
<th>( \varphi_2 )</th>
<th>( \varphi_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 1 \leq \ell &lt; i )</td>
<td>( \ell = i \leq n )</td>
<td>( i &lt; \ell \leq n )</td>
</tr>
<tr>
<td>( \varphi = (i = n + 1) )</td>
<td>( \psi_1 = (a_1^0 = b_1^0) )</td>
<td>( \psi_2 = \text{False} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( \varphi_1 )</th>
<th>( \varphi_2 )</th>
<th>( \varphi_3 )</th>
<th>( \varphi_4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \ell = 1 )</td>
<td>( 2 \leq \ell &lt; i )</td>
<td>( \ell = i )</td>
<td>( i &lt; \ell \leq n )</td>
</tr>
<tr>
<td>( \varphi = (2 \leq i \leq n) )</td>
<td>( \psi_1 = (a_1^0 \leq a_1^1) )</td>
<td>( \psi_2 = (a_2^0 \geq a_2^{−1}) )</td>
<td>( \psi_3 = (a_3^0 = x) )</td>
</tr>
</tbody>
</table>

**Remark:** if \( \varphi \Rightarrow \neg (\exists \ell \varphi_p) \), \( \psi_p \) can be normalized to False:

\[
\forall \ell, \ell \in \emptyset \Rightarrow \text{False}(\ell)
\]
Example of analysis

```plaintext
i := 1;
while (i ≤ n){
   A[i] := B[i];
   i := i+1;
}
```
### Example of analysis

Partition: \( \varphi_1 = (1 \leq \ell < i) \), \( \varphi_2 = (1 \leq \ell = i \leq n) \), \( \varphi_3 = (i < \ell \leq n) \)

\[
\begin{align*}
\text{i := 1;} \\
\text{while (i \leq n)\{} \\
\quad \text{A[i] := B[i];} \\
\quad \text{i := i+1;} \\
\text{\}}
\end{align*}
\]
Example of analysis

Partition: \( \varphi_1 = (1 \leq \ell < i) \), \( \varphi_2 = (1 \leq \ell = i \leq n) \), \( \varphi_3 = (i < \ell \leq n) \)

\[
\begin{align*}
i := 1; \\
\text{while } (i \leq n) \{ \\
A[i] := B[i]; \\
i := i+1;
\}
\end{align*}
\]

<table>
<thead>
<tr>
<th>( \varphi )</th>
<th>( \psi_1 )</th>
<th>( \psi_2 )</th>
<th>( \psi_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>((i = 1))</td>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>
**Example of analysis**

Partition: \( \varphi_1 = (1 \leq \ell < i) \), \( \varphi_2 = (1 \leq \ell = i \leq n) \), \( \varphi_3 = (i < \ell \leq n) \)

\[
egin{align*}
i &:= 1; \\
\text{while} \ (i \leq n) \{ & \\
\quad A[i] &:= B[i]; \\
\quad i &:= i+1; \\
\}
\end{align*}
\]

<table>
<thead>
<tr>
<th>( \varphi )</th>
<th>( \psi_1 )</th>
<th>( \psi_2 )</th>
<th>( \psi_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( i = 1 )</td>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>( i = 1 \leq n )</td>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>
Example of analysis

Partition: \( \varphi_1 = (1 \leq \ell < i) \), \( \varphi_2 = (1 \leq \ell = i \leq n) \), \( \varphi_3 = (i < \ell \leq n) \)

\[
\begin{align*}
i &:= 1; \\
\text{while } (i \leq n) \{ \\
\quad &A[i] := B[i]; \\
\quad &i := i+1;
\}
\end{align*}
\]

\[
\begin{array}{c|c|c|c|c}
\varphi & \psi_1 & \psi_2 & \psi_3 \\
\hline
(i = 1) & \text{False} & \text{True} & \text{True} \\
(i = 1 \leq n) & \text{False} & \text{True} & \text{True} \\
(i = 1 \leq n) & \text{False} & (a_2 = b_2) & \text{True}
\end{array}
\]
Example of analysis

Partition: $\varphi_1 = (1 \leq \ell < i)$, $\varphi_2 = (1 \leq \ell = i \leq n)$, $\varphi_3 = (i < \ell \leq n)$

\[
\begin{array}{cccc}
\text{i := 1;} & \varphi & \psi_1 & \psi_2 & \psi_3 \\
\text{while (i \leq n)} & (i = 1) & \text{False} & \text{True} & \text{True} \\
A[i] := B[i]; & (i = 1 \leq n) & \text{False} & \text{True} & \text{True} \\
i := i+1; & (i = 1 \leq n) & \text{False} & (a_2 = b_2) & \text{True} \\
\}\end{array}
\]
**Example of analysis**

Partition: \( \varphi_1 = (1 \leq \ell < i) \), \( \varphi_2 = (1 \leq \ell = i \leq n) \), \( \varphi_3 = (i < \ell \leq n) \)

<table>
<thead>
<tr>
<th></th>
<th>( \varphi )</th>
<th>( \psi_1 )</th>
<th>( \psi_2 )</th>
<th>( \psi_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( (i = 1) )</td>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>( (1 \leq i \leq n) )</td>
<td>( (a_1 = b_1) )</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>( (i = 2 \leq n + 1) )</td>
<td>( (a_1 = b_1) )</td>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>

\[ \text{A}[i] := \text{B}[i]; \]
\[ \text{i} := 1; \]
\[ \text{while } (i \leq n) \{ \]
\[ \text{A}[i] := \text{B}[i]; \]
\[ \text{i} := \text{i} + 1; \]
\[ \} \]
Example of analysis

Partition: $\varphi_1 = (1 \leq \ell < i)$, $\varphi_2 = (1 \leq \ell = i \leq n)$, $\varphi_3 = (i < \ell \leq n)$

<table>
<thead>
<tr>
<th>$i$</th>
<th>$\varphi$</th>
<th>$\psi_1$</th>
<th>$\psi_2$</th>
<th>$\psi_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i := 1$</td>
<td>$(i = 1)$</td>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>while ($i \leq n$){</td>
<td>$(1 \leq i \leq n)$</td>
<td>($a_1 = b_1$)</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>A[i] := B[i];</td>
<td>$(1 \leq i \leq n)$</td>
<td>($a_1 = b_1$)</td>
<td>($a_2 = b_2$)</td>
<td>True</td>
</tr>
<tr>
<td>i := i+1; }</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example of analysis

Partition: $\varphi_1 = (1 \leq \ell < i)$, $\varphi_2 = (1 \leq \ell = i \leq n)$, $\varphi_3 = (i < \ell \leq n)$

<table>
<thead>
<tr>
<th></th>
<th>$\varphi$</th>
<th>$\psi_1$</th>
<th>$\psi_2$</th>
<th>$\psi_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>i := 1;</td>
<td>(i = 1)</td>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>while (i \leq n){</td>
<td>(1 \leq i \leq n)</td>
<td>(a_1 = b_1)</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>A[i] := B[i];</td>
<td>(1 \leq i \leq n)</td>
<td>(a_1 = b_1)</td>
<td>(a_2 = b_2)</td>
<td>True</td>
</tr>
<tr>
<td>i := i+1;</td>
<td>(2 \leq i \leq n+1)</td>
<td>(a_1 = b_1)</td>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>
Example of analysis

Partition: \( \varphi_1 = (1 \leq \ell < i) \), \( \varphi_2 = (1 \leq \ell = i \leq n) \), \( \varphi_3 = (i < \ell \leq n) \)

<table>
<thead>
<tr>
<th></th>
<th>( \varphi )</th>
<th>( \psi_1 )</th>
<th>( \psi_2 )</th>
<th>( \psi_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>i := 1;</td>
<td>( (i = 1) )</td>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>while (i \leq n){</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A[i] := B[i];</td>
<td>( (1 \leq i \leq n) )</td>
<td>( (a_1 = b_1) )</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>i := i+1;</td>
<td>( (1 \leq i \leq n) )</td>
<td>( (a_1 = b_1) )</td>
<td>( (a_2 = b_2) )</td>
<td>True</td>
</tr>
<tr>
<td>}</td>
<td>( (2 \leq i \leq n+1) )</td>
<td>( (a_1 = b_1) )</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>( (i = n+1) )</td>
<td>( (a_1 = b_1) )</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>
Example of analysis

Partition: $\varphi_1 = (1 \leq \ell < i)$, $\varphi_2 = (1 \leq \ell = i \leq n)$, $\varphi_3 = (i < \ell \leq n)$

\[
\begin{array}{|c|c|c|c|c|}
\hline
i := 1; & \varphi & \psi_1 & \psi_2 & \psi_3 \\
\hline
\text{while } (i \leq n) \{ & (i = 1) & \text{False} & \text{True} & \text{True} \\
A[i] := B[i]; & (1 \leq i \leq n) & (a_1 = b_1) & \text{True} & \text{True} \\
i := i + 1; & (1 \leq i \leq n) & (a_1 = b_1) & (a_2 = b_2) & \text{True} \\
\} & (2 \leq i \leq n + 1) & (a_1 = b_1) & \text{True} & \text{True} \\
(i = n + 1) & (a_1 = b_1) & \text{False} & \text{True} & \text{False} \\
\hline
\end{array}
\]

$\forall \ell, 1 \leq \ell \leq n, A[\ell] = B[\ell]$
Some results

- **Insertion sort**
  \[ \forall \ell, (2 \leq \ell \leq n) \Rightarrow (A[\ell - 1] \leq A[\ell]) \]

- **Find (QuickSort segmentation)**
  \[
  (i = n \land \forall \ell, (1 \leq \ell \leq n - 1) \Rightarrow (A[\ell] \leq A[i])
  
  \lor 
  i = 1 \land \forall \ell, (2 \leq \ell \leq n) \Rightarrow (A[i] < A[\ell])
  
  \lor 
  1 < i < n \land \forall \ell, (1 \leq \ell \leq i - 1) \Rightarrow (A[\ell] \leq A[i])
  
  \land (i + 1 \leq \ell \leq n) \Rightarrow (A[i] < A[\ell])
  \]
Future work

- improve the implementation
Future work

- improve the implementation
- more general programs ("for" loops with steps, recursivity...)

M. Péron, N. Halbwachs (Verimag/CNRS)
Future work

- improve the implementation
- more general programs ("for" loops with steps, recursivity...)
- more general properties (non convex slices)
Future work

- improve the implementation
- more general programs ("for" loops with steps, recursivity...)
- more general properties (non convex slices)
- multi-dimensional arrays?
Future work

- improve the implementation
- more general programs ("for" loops with steps, recursivity...)
- more general properties (non convex slices)
- multi-dimensional arrays?
- generalization to function properties?
Future work

- improve the implementation
- more general programs ("for" loops with steps, recursivity...)
- more general properties (non convex slices)
- multi-dimensional arrays?
- generalization to function properties?
- properties about (multi-)sets of array values