Southwest State University
Department of Computer Sciences

COMPARISON OF SEQUENTIAL METHODS FOR GETTING SEPARATIONS OF PARALLEL LOGIC CONTROL ALGORITHMS USING VOLUNTEER COMPUTING

Eduard I. Vatutin
Sergey Yu. Valyaev
Vitaly S. Titov

BOINC:FAST’15
Petrozavodsk, 2015
Application area of logic control systems

Implements parallel logic control algorithms for logic control of the operation part of some control object.

- digital circuitry – coprocessors, accelerators (performance);
- production complexes – automated assembly cells, CNC machines (reliability).
Control object and Logic Control System (LCS)

Parallel logic control algorithm

Development

Logic control system

Logic conditions
\[ X = x_1, x_2, \ldots, x_N \]

Control object

Microoperations
\[ Y = y_1, y_2, \ldots, y_M \]

Problems:
1. Separation
2. Allocation
3. Fault tolerance
4. Routing
5. Collective interactions
   etc.
Statement of a problem

\[ \bigcup_{i=1}^{H} A_i = A^0, \quad A_i \neq \emptyset, \quad A_i \cap A_j = \emptyset, \quad i, j = 1, H, \quad i \neq j, \]

\[ \neg (a_i \wedge a_j) \quad \forall a_i, a_j \in A_k, \quad i \neq j, \quad k = 1, H, \]

\[ W(A_i) \leq W_{\text{max}}, \quad |X(A_i)| \leq X_{\text{max}}, \quad |Y(A_i)| \leq Y_{\text{max}}, \quad i = 1, H. \]

Discrete combinatorial optimization problem
NP-hard problem
Multicriteria optimization problem
Quality criteria

\[ H \rightarrow \min \]

\[ Z_1 = \sum_{i=1}^{H} \sum_{j=1, j \neq i}^{H} \alpha(A_i, A_j) \rightarrow \min \]

\[ Z_2 = \sum_{i=1}^{H} \sum_{j=1, j \neq i}^{H} \delta(A_i, A_j) \rightarrow \min \]

\[ Z_3 = \sum_{i=1}^{H} \left| X(A_i) - X(A^0) \right| \rightarrow \min \]

\[ Z_4 = \sum_{i=1}^{H} \left| Y(A_i) - Y(A^0) \right| \rightarrow \min \]
Example of separation (given with using PAE program system)
Methods of getting separations overview

Methods of getting separations:

- **Brute Force** (NP-hard problem, N<10 only);
- **S.I. Baranov method** (1984, greedy approach);
- **Greedy with sequential building of blocks and using adjacent vertices** (2013, greedy with additional constraints);
- A.D. Zakrevsky method (coloring of special graph);
- **Parallel-sequential method** (1997–Now, special method with transformations of graph-schemes of parallel algorithms);
- Well known discrete combinatorial optimization methods (stochastic methods: random search, directed random search, ACO, BC, GA, SA, etc.).

Quality of decisions, time and memory costs, complexity and features of practical implementation are significantly differ!

Which method is the best?
Brief characteristics of methods: greedy approaches

\[
f(a_i, A^{(j)}) = k_1 \frac{\Delta X(A^{(j)}, A^{(j+1)})}{X_{\text{max}} - \left| X(A^{(j+1)}) \right| + 1} + k_2 \frac{\Delta Y(A^{(j)}, A^{(j+1)})}{Y_{\text{max}} - \left| Y(A^{(j+1)}) \right| + 1} + k_3 \frac{\Delta W(A^{(j)}, A^{(j+1)})}{W_{\text{max}} - W(A^{(j+1)}) + 1},
\]

\[
M_R = \begin{pmatrix}
- & \varphi & \varphi & \varphi & \varphi & \varphi & \varphi & \varphi & \varphi \\
\nu, \varphi & - & \varphi & \varphi & \varphi & \varphi & \varphi & \varphi & \varphi \\
\nu, \varphi & \nu, \varphi & - & \psi & \psi & \psi & \psi & \psi & \psi \\
\nu, \varphi & \nu, \varphi & \psi & - & \Psi & \psi & \psi & \psi & \psi \\
\nu, \varphi & \nu, \varphi & \psi & \psi & \nu, \varphi & - & \psi & \psi & \psi \\
\nu, \varphi & \nu, \varphi & \psi & \psi & \nu, \varphi & \psi & - & \Psi & \psi \\
\nu, \varphi & \nu, \varphi & \psi & \psi & \nu, \varphi & \psi & \psi & - & \varphi \\
\nu, \varphi & \nu, \varphi & \psi & \psi & \nu, \varphi & \psi & \psi & \psi & \varphi \\
\nu, \varphi & \nu, \varphi & \psi & \psi & \nu, \varphi & \nu, \varphi & \nu, \varphi & \nu, \varphi & \nu, \varphi & - & \varphi
\end{pmatrix}

O(N) or O(N^3)?
Brief characteristics of methods: parallel-sequential approach
Equivalent transformation of graph-scheme using tree of fragments
Generating test data – graph-schemes with random structure
**Working with sections**

\[ S^u(\Omega_j) = \{ S_{s_1}, S_{s_2}, \ldots, S_{s_p} \} \]

\[ R^u_j \subseteq \Omega_j, \quad j = 1, P \]

\[ S^d(\Omega_j) = \{ S_{d_1}, S_{d_2}, \ldots, S_{d_q} \} \]

\[ R^d_k \subseteq \Omega_j, \quad k = 1, Q \]

\[ \Omega = a_5 \cdot a_7 \cdot (a_7 \mid a_8 \mid (a_9 \cdot a_{10})) \]

\[ S^u = \{ S_4, S_7 \} \quad \Omega = a_2 \cdot a_5 \cdot a_6 \cdot \left( a_7 \mid a_8 \mid \frac{(a_9 \cdot a_{10})}{(a_7 \cdot a_{11})} \right) = a_2 \cdot a_4 \cdot (a_7 \mid a_8 \mid a_{22}) \]

\[ S^d = \{ S_5 \} \quad \Omega = a_5 \cdot a_2 \cdot a_6 \cdot \left( a_7 \mid a_8 \mid \frac{(a_9 \cdot a_{10})}{(a_5 \cdot a_{11})} \right) = a_5 \cdot a_2 \cdot a_6 \cdot (a_7 \mid a_8 \mid a_{23}) \]

\[ \mathcal{R} = \left\{ a_0, a_1, a_2 \cdot a_4 \cdot a_3, a_2 \cdot a_4 \cdot (a_7 \mid a_8 \mid a_{22}), a_5 \cdot a_2 \cdot a_6 \cdot (a_7 \mid a_8 \mid (a_9 \cdot a_{10})), a_5 \cdot a_2 \cdot a_6 \cdot (a_7 \mid a_8 \mid a_{23}), a_5 \cdot a_2 \cdot a_6 \cdot a_{11}, a_5 \cdot a_2 \cdot a_4 \cdot a_{13}, a_5 \cdot a_1 \cdot a_8 \cdot (a_14 \mid a_{15}), a_5 \cdot a_1 \cdot a_8 \cdot a_{16}, a_{19} \cdot a_{18}, a_{20}, a_{21} \right\} \]

Determining parallelism degree and base section (most wide part) at polynomial time!
Parallel blocks building using weight function and conclusion tables

\[
t(\tilde{\Omega}_i, A_j) = \begin{cases} 
- \text{"-", } \exists a_m \in \tilde{\Omega}_i, a_n \in A_j : a_m \omega a_n, \\
+ \text{"+', } (W(\tilde{\Omega}_i) + W(A_j) > W_{\text{max}}) \lor (|X(\tilde{\Omega}_i) \cup X(A_j)| > n_{\text{DU}}) \lor \\
\lor (|Y(\tilde{\Omega}_i) \cup Y(A_j)| > n_{\text{MO}}), \\
K_i^Y |Y(\tilde{\Omega}_i) \cap Y(A_j)| - K_2^Y |Y(\tilde{\Omega}_i) \setminus Y(A_j)| + \\
+ K_i^X |X(\tilde{\Omega}_i) \cap X(A_j)| - K_2^X |X(\tilde{\Omega}_i) \setminus X(A_j)| - \\
-(K_\alpha \Delta Z_\alpha + K_\beta \Delta Z_\beta) + K_\psi |\psi(\tilde{\Omega}_i)| |\psi(A_j)| \text{ иначе,}
\end{cases}
\]

\[
\Theta(\Upsilon(\Omega_{k+1}), \Gamma_k) = \begin{bmatrix} \tilde{\Omega}_1 & \tilde{\Omega}_2 & \ldots & \tilde{\Omega}_{N_T} \\
A_1 & A_2 & \ldots & A_{N_T} \end{bmatrix}
\]

\[
\sum_{\{\Omega_j, A_j\} \in \Theta(\Upsilon(\Omega_{k+1}), \Gamma_k)} t(\Omega_j, A_j) \rightarrow \text{max}
\]

Very similar to assignment problem
Quality analysis: parameters space and experiments

- fixed parameters experiments (2007), 10 minutes of computing time;
- variable value of 1 parameter (2008), 5 hours of computing time;
- variable values of 2 parameters (2010–Now), 430 years of computing time (with using BOINC).
Parallel computing with using grid

- 2000+ volunteers;
- 1000+ hosts;
- real performance – 2 – 3 TFLOP/s;
- time gain – 558 times (vs Core 2 Duo @ 1.86 GHz using 24/7);
- volume of raw experimental data – 500+ GB.
Postprocessing

\[
\gamma_x(F_i) = \frac{\sum_{j=1}^{K} Z_x(Sep_{F_i}(A_j^0))}{K}, \quad x \in \{H, X, Y, \alpha, \delta, J\}
\]

\[
\rho_x(F_i) = \frac{\sum_{j=1}^{K} \mu(Z_x(Sep_{F_i}(A_j^0)))}{K}
\]

\[
\mu(Z_x(Sep_{F_i}(A_j^0))) = \begin{cases} 
1, & Z_x(Sep_{F_i}(A_j^0)) = \\
\min_{l=1,N_F} Z_x(Sep_{F_i}(A_j^0)), & \text{otherwise}
\end{cases}
\]

\[
\gamma_H = \frac{\sum_{i=1}^{K} Z_H(Sep(A_i^0))}{K}
\]

\[
\rho_X = \frac{\sum_{i=1}^{K} \mu(Z_X(Sep(A_i^0)))}{K}
\]
Postprocessing of raw data

1. Building minimal map, calculating of average maps (for each experiment)

\[ M_{\text{min}} = \min_{i=1, n} M_{F_i} \] – (n-1) comparisons, for each: reading 105 GB from HDD, writing 26 GB to HDD, 26 hours

2. Building probability maps (for each experiment)

\[ M_{\rho}(F_i) = \mu\left(M_{F_i}, M_{\text{min}}\right) \] – n comparisons, for each: reading 105 GB from HDD, writing 10 MB to HDD, 20 hours

3. Building prefer map (each for all experiments)

And also:

- preparing source data (getting from server, splitting by experiments) – working with 1 000 000+ files with 200 KB size (NTFS is very slow);
- manual analysis of given maps, articles publishing.
Example: probability maps for adjacent greedy strategy
Example: deviation of average quality from optimum
**Example: prefer maps**

red – S.I. Baranov method,  
green – parallel-sequential method,  
blue – greedy adjacent strategy,  
yellow – random search.

Observed strong zone dependence of quality of decisions for different methods from area of parameters space!

What method is best? Answer depends from area of parameters space!
Decreasing quality during increasing power of constraints
### Recommendations for LCS hardware structure

<table>
<thead>
<tr>
<th>Number of controllers (H) within multicontroller without redundancy</th>
<th>Average number of vertices (N) with graph-scheme</th>
<th>Maximal constraint value (without decreasing of integral criterion (J) value)</th>
<th>Maximal constraint value (5% decreasing of integral criterion (J) value)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3×3 = 9</td>
<td>50</td>
<td>30</td>
<td>34</td>
</tr>
<tr>
<td>4×4 = 16</td>
<td>110</td>
<td>49</td>
<td>61</td>
</tr>
<tr>
<td>5×5 = 25</td>
<td>200</td>
<td>64</td>
<td>84</td>
</tr>
<tr>
<td>6×6 = 36</td>
<td>312</td>
<td>76</td>
<td>105</td>
</tr>
<tr>
<td>7×7 = 49</td>
<td>450</td>
<td>86</td>
<td>120</td>
</tr>
<tr>
<td>7×8 = 56</td>
<td>535</td>
<td>89</td>
<td>126</td>
</tr>
</tbody>
</table>

- Hardware complexity of LCS can be significantly reduced by small (5%) decreasing quality of separations;
- LCS with many simple controllers is preferable.
Prospects for future research

- computation experiments with different heuristic methods at separation problem (variations of stochastic approaches, variations of greedy approaches, Hungarian algorithm using, combinations of methods, early discarding of bad decisions, etc.);
- different problems during LCS design;
- graph theory problems (shortest paths, Hamiltonian paths and loops, isomorphism, coloring);
- schedules building.

\[
f(\Theta) = \frac{K_w^{(t)} t_w (\Theta)}{2Dw^{(t)}_\text{max} T} + \frac{K_w^{(s)} s_w (\Theta)}{2Dw^{(s)}_\text{max} S} + \frac{K_{wt}^{(t)} t_{wt} (\Theta)}{2Dp^{(t)}_\text{max} T} + \frac{K_{wt}^{(s)} s_{wt} (\Theta)}{2Dp^{(s)}_\text{max} S} + \]
\[
+ \frac{K_w^{(a)} a_w (\Theta)}{2DPA} + \frac{K_{wa}^{(w)} w_a (\Theta)}{2DT} + \frac{K_g^{(t)} t_g (\Theta)}{2DT} + \]
\[
+ \frac{K_{p_{\text{min}}}^{(t)} t_{p_{\text{min}}} (\Theta)}{2DT} + \frac{K_{p_{\text{max}}}^{(t)} t_{p_{\text{max}}} (\Theta)}{2DT} + \frac{K_{p_{\text{max}}}^{(s)} s_{p_{\text{max}}} (\Theta)}{2DS} + \frac{K_{p_{\text{max}}}^{(s)} s_{p_{\text{max}}} (\Theta)}{2DS} + \]
\[
+ \frac{K_{s_{\text{min}}}^{(s)} s_{s_{\text{min}}} (\Theta)}{2DS} + \frac{K_{s_{\text{max}}}^{(s)} s_{s_{\text{max}}} (\Theta)}{2DS} + \frac{K_{m_{\text{max}}}^{(t)} t_{m_{\text{max}}} (\Theta)}{2DT \left( \text{max } m_{ij} + 1 \right)} + \frac{K_{m_{\text{max}}}^{(s)} s_{m_{\text{max}}} (\Theta)}{2DS \left( \text{max } m_{ij} + 1 \right)} \]
The End.
Thanks!

The authors would like to thank all volunteers who took part in the calculation within the distributed computing project Gerasim@Home!

WWW: http://evatutin.narod.ru
E-mail: evatutin@rambler.ru
LJ: http://evatutin.livejournal.com
Skype: evatutin
vk, ok, facebook