Johnson's procedure: mechanization and parallelization

M. Filali N. Zaidi

IRIT-CNRS ; Lycée Blaise Pascal

RTSOPS
2016 July, 5th
ECRTS
Toulouse

ACADIE

Real-time

Ubiquitous and critical domain :

- Application level :
  - Transport : avionics, space, trains, cars, ...
  - Transactions : travel, business, ...
- Processor level :
  - Multicore architectures.
  - Instruction scheduling.

Real-Time Challenges

How to reuse the huge real-time scheduling knowledge ?

- C1. Critical safety constraint
- C2. How to adapt to multicores ?

Context

- Method space DO-178C Technology space
  - Model-based development and verification DO-331
  - OO Technology and related techniques DO-332
  - Formal methods DO-333

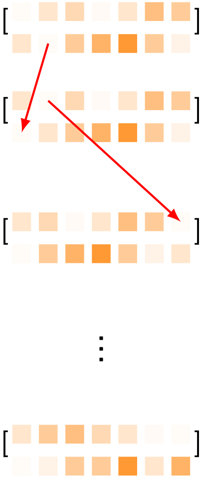
Generic problem(wikipedia) : *Jobs consisting of multiple operations. The basic form of the problem of scheduling jobs with multiple (M) operations, over M machines, such that all of the first operations must be done on the first machine, all of the second operations on the second, etc., and a single job cannot be performed in parallel, is known as the open shop scheduling problem.*

Application :

- computer science.
- instruction scheduling.
- railway domain.

- Folklore of real-time algorithms (1954), presented (informally) in most of real-time textbooks.
- text :
  starting from a list of jobs characterized by the durations required sequentially and exclusively on two machines, the optimal order is found as follows :
  iteratively look for the minimum over the first durations, if this minimum is less than the minimum over the last durations, put this job first, otherwise put this job last.

Johnson's procedure
Graphical presentation



Makespan definition : ms

$$\mathbf{ms}_i \in (\mathbb{N}\times\mathbb{N})\text{list} \times (\mathbb{N}\times\mathbb{N}) \to \mathbb{N}\times\mathbb{N}$$
$$(l, i) \ \ (*\ \text{job list, initial availability}\ *) \mapsto$$
$$\mathbf{reduce}(\lambda(a_1,a_2) : \lambda(d_1,d_2) : (a_1+d_1, \max(a_1+d_1, a_2)+d_2)), l, i)$$

and $\mathbf{ms}(l) = \mathbf{ms}_i(l,(0,0))$.

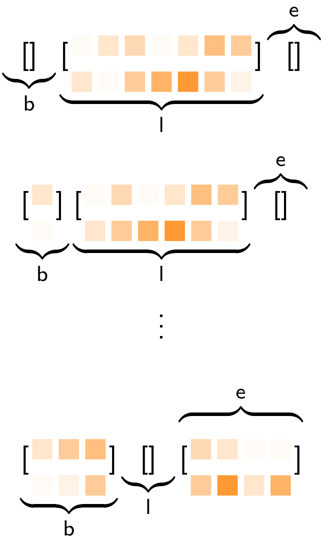
Characterizing properties :

(J1) **J** is a total function : $\mathbf{J} \in (\mathbb{N}\times\mathbb{N})\text{list} \to (\mathbb{N}\times\mathbb{N})\text{list}$.

(J2) **J** is conservative : **J** does not create or loose jobs. **J** generates a permutation : $\forall l.\ \mathbf{J}(l) \simeq l$.

(J3) **J** is optimal :
$\forall l\ l'.\ l \sim l' \Rightarrow \mathbf{ms}(\mathbf{J}(l))_1 \leq \mathbf{ms}(l')_1 \wedge \mathbf{ms}(\mathbf{J}(l))_2 \leq \mathbf{ms}(l')_2$.

Use of the auxiliary variables : b, e.



termination when $l = []$, the result is $b++e$.

Reasoning

- basic transitions : $(b, l, e) \rightsquigarrow (b', l', e')$

$$\frac{\begin{array}{ll} & l \neq [] \wedge \min_1(l) \leq \min_2(l) \\ \wedge & b' = b ++ [J_1 l] \wedge l' = \text{remove1}(J_1(l), l) \wedge e' = e \end{array}}{\begin{array}{ll} & l \neq [] \wedge \neg(\min_1(l) \leq \min_2(l)) \\ \wedge & b' = b \wedge l' = \text{remove1}(J_2(l), l) \wedge e' = (J_2 l)\sharp e \end{array}}$$

- invariants
  - $\text{inv\_permutation}(l_i, b, l, e) = (l_i \simeq (b ++ l ++ e))$
  - $\text{inv\_partition}(b, l, e) = \begin{array}{ll} & b = [(x, y) \leftarrow b.\ x \leq y] \\ \wedge & \forall (x, y) \in b.\ \forall (x', y') \in l.\ x \leq x' \\ \wedge & \forall (x, y) \in l.\ \forall (x', y') \in e.\ y \geq y' \\ \wedge & e = [(x, y) \leftarrow e.\ \neg x \leq y] \\ \wedge & \text{inc}_1(b) \\ \wedge & \text{dec}_2(e) \end{array}$

Johnson's procedure characterization

$$inv\_J(L) = \begin{array}{ll} & L = \begin{array}{c} [(x,y) \textbf{ for } (x,y) \textbf{ in } L \textbf{ if } x \geq y] \\ ++ \\ [(x,y) \textbf{ for } (x,y) \textbf{ in } L \textbf{ if } \neg x \geq y] \end{array} \\ \wedge & \mathrm{inc}_1([(x,y) \textbf{ for } (x,y) \textbf{ in } L \textbf{ if } x \geq y]) \\ \wedge & \mathrm{dec}_2([(x,y) \textbf{ for } (x,y) \textbf{ in } L \textbf{ if } \neg x \geq y]) \end{array}$$

Thanks to the invariant, we show : $\forall l.\ \mathrm{inv\_J}(J(l))$

Optimality :

$$\forall l\ l'.\ l \sim l' \Rightarrow \mathbf{ms}(\mathbf{J}(l))_1 \leq \mathbf{ms}(l')_1 \wedge \mathbf{ms}(\mathbf{J}(l))_2 \leq \mathbf{ms}(l')_2$$

two steps (lemmas) :

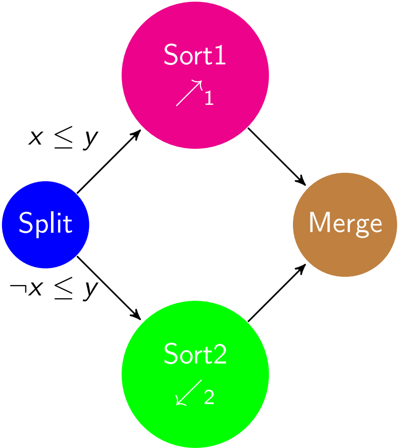
- $ms(J(l), i) \leq ms(l, i)$
  Usual invariance proof : each transition decreases the makespan.
- $l \simeq l' \wedge \text{inv\_J}(l) \wedge \text{inv\_J}(l') \Rightarrow ms(l, i) = ms'(l, i)$
  Reasoning simultaneously over 2 lists (double induction).

Mechanization

- Use of the Isabelle-HOL assistant theorem prover.
- formalization of the basic definitions (e.g. ms) and procedures (J).
- Proof of the invariants, optimality property.
  - basic theorems concern permutations and swapping properties.
  - proofs are difficult because the scheduled list has bad algebraic properties.

Parallelization

$$
\begin{aligned}
\text{inv\_J(L)} = \quad & L = \begin{array}{c} [(x,y)\ \textbf{for}\ (x,y)\ \textbf{in}\ L\ \textbf{if}\ x \geq y] \\ +\!+ \\ [(x,y)\ \textbf{for}\ (x,y)\ \textbf{in}\ L\ \textbf{if}\ \neg x \geq y] \end{array} \\
\wedge \quad & \text{inc}_1([(x,y)\ \textbf{for}\ (x,y)\ \textbf{in}\ L\ \textbf{if}\ x \geq y]) \\
\wedge \quad & \text{dec}_2([(x,y)\ \textbf{for}\ (x,y)\ \textbf{in}\ L\ \textbf{if}\ \neg x \geq y])
\end{aligned}
$$



Conclusion

- Presented work :
  - Johnson's procedure, mechanization
    $\rightsquigarrow$ use of formal methods for addressing the critical safety constraint.
  - Parallelization
    $\rightsquigarrow$ adptation of existing real-time kowledge to multicore architectures
    (hint : invariant, properties of the sequential algorithm).
- Future work : real-time algorithms and their parallelization.
  - Disjunctive constraint.
  - Precise real-time analysis (link with timed automata).

$$
\begin{aligned}
J_a\ b\ l\ e = \ & \textbf{if}\ l = []\ \textbf{then}\ (b, [], e) \\
& \textbf{else if}\, Min_1 l \leq Min_2 l\ \textbf{then} \\
& \quad J_a\ (b @ [J_1 l])\ (remove1 (J_1 l) l)\ e \\
& \textbf{else}\ J_a\ b\ (remove1 (J_2 l) l)\ ((J_2 l) \sharp e
\end{aligned}
$$

$$
\begin{aligned}
J\ l = \ & \textbf{let}\ (b', l', e') = J_a\ []\ l\ []\ \textbf{in} \\
& \quad b' @ e'
\end{aligned}
$$