Algorithms for LMAS Performance Assessment

The following lists algorithms as used in the following publication:

"A priori performance assessment of line-less mobile assembly systems"

Authors: R.H. Schmitt, G. Hüttemann and S. Münker

DOI: https://doi.org/10.1016/j.cirp.2021.04.059

Cite as: Robert H. Schmitt, Guido Hüttemann, Sören Münker: A priori performance assessment of line-less mobile assembly systems. CIRP Annals, Volume 70, Issue 1, 2021,Pages 389-392

Algorithms are based on work by AKYILDIZ (Akyildiz, I.an F.: 'Mean value analysis of closed queuing networks with Erlang service time distributions'. *Computing* (1987), vol. 39(3): pp.219-232.).

Algorithm 1

All equation references are with regard to the reference above.

Algorithm 1: Extended Mean Value Analysis

Result: Determined KPI for tested configuration
Input: $T_{(P,PC)}$, $\mathbf{f}_P$, $\mathbf{C}_{(SC,PC)}$, $\mathbf{D}$, $\mathbf{RC}$, U, n_{TS}, v_{TS}
Output: λ_{tot}, LT_j, w_t^Q, n_t^Q
begin
    Initialization (U=0)
    Set $\mathbf{N}_{(P,S)}(U=0)$ to zero
    Set $\mathbf{N}^{proc}_{(P,S)}(U=0)$ to zero
    Set $\mathbf{W}_{(P,S)}(U=0)$ to $\mathbf{T}_{(P,S)}$
    for $U=1;\ U \leq U_{max};\ U=U+1$ **do**
        Calculate $\mathbf{u}_P(U)$ (Eq. 11)
        Calculate $\mathbf{W}_{(P,S)}(U)$ (Eq. 12)
        Calculate $\breve{}_P(U)$ (Eq. 13)
        Calculate $\mathbf{X}_{(P,S)}(U)$ (Eq. 14)
        Calculate $\mathbf{N}_{(P,S)}(U)$ (Eq. 15)
        Calculate $\mathbf{W}^{proc}_{(P,S)}(U)$ (Eq. 16)
        Calculate ρ_t for all s_t (Eq. 19)
        if *any* $\rho_t > 1$ **then**
            Set $\rho_t = 0.9999$
            Calculate variables according to Algorithm 2
        end
    end
    Calculate λ_{tot} (Eq. 17)
    Calculate LT_j for all p_j (Eq. 18)
    Calculate w_t^Q for all s_t (Eq. 20)
    Calculate n_t^Q for all s_t (Eq. 21)
end

Algorithm 2

Additional equations:

$$\lambda_j(U) = \frac{0.9999 * \mathbf{f}_P}{\sum_{t=1}^{T} t_{jt}(U) * q_{jt}} \tag{0.1}$$

$$x_{jt}(U) = \lambda_j(U) * q_{jt} \tag{0.2}$$

$$n^*_{jt}(U) = x_{jt}(U) * w_{jt}(U) \tag{0.3}$$

$$n_{rest}(U) = U - \sum_{j=1}^{J} \sum_{t=1}^{T} n_{jt} \tag{0.4}$$

$$n_{jt}(U) = n^*_{jt}(U) \ + \ n_{rest}(U) \tag{0.5}$$

$$w_{jt}(U) = \frac{u_j(U)}{\lambda_j(U) * q_{jt}} \tag{0.6}$$

Algorithm 2: Boundary algorithm after Akyildiz

Result: Re-determine variable for overloaded server s_t

Input: Input from Algortihm 1

Output: $\lambda_j(U)$, $x_{jt}(U)$, $n^*_{jt}(U)$, $n_{rest}(U)$, $n_{jt}(U)$, $w_{jt}(U)$

begin

  Calculate new throughput $\lambda_j(U)$ (Eq: 0.1)

  Calculate new flows $x_{jt}(U)$ (Eq: 0.2)

  Calculate theoretic number of stations $n^*_{jt}(U)$ (Eq: 0.3)

  Calculate missing number of jobs $n_{rest}(U)$ (Eq: 0.4)

  Calculate actual number of stations $n_{jt}(U)$ (Eq: 0.5)

  Calculate new dwell times $w_{jt}(U)$ (Eq: 0.6)

end