4 Approximative Analyse von Produktformwarteschlangennetzen

- Approximative Mittelwertanalyse (Bard-Schweitzer-Methode):

  - ➤ Ausgangsgleichung: Ankunftstheorem:

$$\overline{T}_i(K) = \frac{1}{\mu_i} \cdot \left[1 + \overline{K}_i(K-1)\right] , \qquad i = 1, \dots , N$$

  - ➤ Iteration nicht über k =1, 3, 4, ... , K sondern mithilfe der Approximation:

$$\overline{K}_i(K-1) = \frac{K-1}{K} \cdot \overline{K}_i(K)$$

  - ➤ Mit dem Startwerten:

$$\overline{K}_1(K) = \overline{K}_2(K) = \cdots = \overline{K}_N(K) = \frac{K}{N}$$

◆ Algorithmus:

- ***Schritt*** 1: Initialisierung:

$$\overline{K}_1(K) = \overline{K}_2(K) = \cdots = \overline{K}_N(K) = \frac{K}{N}$$

- ***Schritt*** 2: Approximation:

$$\overline{K}_i(K-1) = \frac{K-1}{K} \cdot \overline{K}_i(K) \qquad i = 1, 2, \ldots, N$$

- ***Schritt*** 3: Mittlere Antwortzeit:

$$\overline{T}_i(K) = \frac{1}{\mu_1}\left[1 + \overline{K}_i(K-1)\right] \qquad i = 1, \cdots, N \qquad \text{Typ 1,2,4}$$

$$\overline{T}_i(K) = \frac{1}{\mu_i} \qquad i = 1, 2, \cdots, N \qquad \text{Typ 3}$$

- ***Schritt*** 4: Durchsatz:

$$\lambda(K) = \frac{K}{\sum_{i=1}^{N} e_i \overline{T}_i(K)}$$

- ***Schritt*** 5: Mittlere Anzahl der Aufträge:

$$\overline{K}_i(K) = \overline{T}_i(K)\lambda(K)e_i \quad i = 1, 2, \cdots, N$$

- ***Schritt*** 6: Abbruchbedingung:

$$\max_i \left| \overline{K}_i^{(n)}(K) - \overline{K}_i^{(n-1)}(K) \right| \quad < \epsilon$$

Ansonsten Sprung nach ***Schritt*** 2

◆ Beispiel: WS-Netz mit $K = 6$ und $\varepsilon = 0.06$ und:

i	e_i	$1/\mu_i$	m_i
1	1	0.02	1
2	0.4	0.2	1
3	0.2	0.4	1
4	0.1	0.6	1

- ***Schritt*** 1: Initialisierung:

$$\overline{K}_1(K) = \overline{K}_2(K) = \overline{K}_3(K) = \overline{K}_4(K) = \frac{K}{N} = \underline{1.5}$$

- ***Schritt*** 2: Approximation ($K = 6$):

$$\overline{K}_1(K-1) = \frac{K-1}{K} \cdot \overline{K}_1(K) = \underline{1.25}\,,$$
$$\overline{K}_2(K-1) = \overline{K}_3(K-1) = \overline{K}_4(K-1) = \underline{1.25}\,.$$

- ***Schritt*** 3: Mittlere Antwortzeit:

$$\overline{T}_1(K) = \frac{1}{\mu_1}\left[1 + \overline{K}_1(K-1)\right] = \underline{0.045}\,, \quad \overline{T}2(K) = \frac{1}{\mu_2}\left[1 + \overline{K}_2(K-1)\right] = \underline{0.45}\,,$$

$$\overline{T}_3(K) = \frac{1}{\mu_3}\left[1 + \overline{K}_3(K-1)\right] = \underline{0.9}\,, \qquad \overline{T}_4(K) = \frac{1}{\mu_4}\left[1 + \overline{K}_4(K-1)\right] = \underline{1.35}\,.$$

- ***Schritt*** 4: Durchsatz:

$$\lambda(K) = \frac{K}{\sum_{i=1}^{4} e_i \overline{T}_i(K)} = \underline{11.111}$$

- ***Schritt*** 5: Mittlere Anzahl der Aufträge:

$$\overline{K}_1(K) = \overline{T}_1(K)\lambda(K)e_1 = \underline{0.5}\,, \quad \overline{K}_2(K) = \overline{T}_2(K)\lambda(K)e_2 = \underline{2}\,,$$

$$\overline{K}_3(K) = \overline{T}_3(K)\lambda(K)e_3 = \underline{2}\,, \qquad \overline{K}_4(K) = \overline{T}_4(K)\lambda(K)e_4 = \underline{1.5}\,.$$

- ***Schritt*** 6: Abbruchbedingung:

$$\max_i \left| \overline{K}_i^{(1)}(K) - \overline{K}_i^{(0)}(K) \right| = \underline{1} \quad > 0.06\,.$$

2. Iteration:

- ***Schritt*** 2: Approximation:

$$\overline{K}_1(K-1) = \frac{K-1}{K} \cdot 0.5 = \underline{0.417}\,,$$
$$\overline{K}_2(K-1) = \underline{1.667}\,, \quad \overline{K}_3(K-1) = \underline{1.667}\,, \quad \overline{K}_4(K-1) = \underline{1.25}\,.$$

Nach 4 Iterationen:

- ***Schritt*** 6: Abbruchbedingung:

$$\max_i \left| \overline{K}_i^{(4)}(K) - \overline{K}_i^{(3)}(K) \right| = \underline{0.053} \quad < 0.06\,.$$

➤ Endgültige Ergebnisse:

Node:	1	2	3	4
Mean response time $\overline{\boldsymbol{T}}_{\boldsymbol{i}}$	0.024	0.573	1.145	1.240
Throughput $\boldsymbol{\lambda_i}$	9.896	3.958	1.979	0.986
Mean number of jobs $\overline{\boldsymbol{K}}_{\boldsymbol{i}}$	0.239	2.267	2.267	1.240
Utilization $\boldsymbol{\rho_i}$	0.198	0.729	0.729	0.594

➤ Vergleich mit exakten Ergebnissen (Mittelwertanalyse):

Node	1	2	3	4
Mean response time $\overline{\boldsymbol{T}}_{\boldsymbol{i}}$	0.025	0.570	1.140	1.244
Throughput $\boldsymbol{\lambda_i}$	9.920	3.968	1.984	0.992
Mean number of jobs $\overline{\boldsymbol{K}}_{\boldsymbol{i}}$	0.244	2.261	2.261	1.234
Utilization $\boldsymbol{\rho_i}$	0.198	0.794	0.794	0.595

➤ Bewertung:

- Genauigkeit f ür die meisten Anwendungen ausreichend
  (Im Mittel ca. 6% Abweichung)

- Keine M/M/m-Knoten

- Weniger Rechenzeit und weniger Speicherplatz als **MVA**

- Auch bei mehreren Auftragsklassen anwendbar

- Bessere Genauigkeit und M/M/m-Knoten beim **SCAT**- Algorithmus

➤ **SCAT** (Self-Correction Approximation Technique):

- Mehrfache Anwendung des BS-Algorithmus'
- Sehr gute Genauigkeit, M/M/m-Knoten, wenig Speicherplatz und Rechenzeit
- Kern des SCAT-Algorithmus:
  - Hilfsfunktionen:

    $F_i(K) = \overline{K}_i(K)/K \text{ und } D_i(K) = F_i(K\text{-}1) - F_i(K)$

  - Verbesserte Approximation:

    $\overline{K}_i(K\text{-}1) = (K\text{-}1)(F_i(K) + D_i(K))$

  - D_i: Korrekturfunktion, wird iterativ verbessert
  - $D_i = 0 \rightarrow$ BS- Algorithmus

Summationsmethode:

Grundlegende Gesetzmäßigkeiten:

- Mittlere Anzahl der Aufträge: $\overline{K}_i = f_i(\lambda_i)$

mit:

$$f_i(\lambda_i) = \overline{K}_i = \begin{cases} \dfrac{\rho_i}{1 - \dfrac{K-1}{K}\rho_i}, & \text{Type-1,2,4 } (m_i = 1), \\[2ex] m_i\rho_i + \dfrac{\rho_i}{1 - \dfrac{K-m_i-1}{K-m_i}\rho_i} \cdot P_{m_i}, & \text{Type-1 } (m_i > 1), \\[2ex] \dfrac{\lambda_i}{\mu_i}, & \text{Type-3.} \end{cases}$$

- Systemgleichung:

$$\sum_{i=1}^{N} \overline{K}_i = \sum_{i=1}^{N} f_i(\lambda_i) = K$$

◆ Algorithmus:

- ***Schritt*** 1: Initialisierung:

  Untere Grenze des Durchsatzes: $\lambda_u = 0$

  Obere Grenze des Durchsatzes: $\lambda_o = \min_i \left\{ \frac{\mu_i m_i}{e_i} \right\}$

- ***Schritt*** 2: Intervallschachtelung

  ➤ *Schritt* 2.1: Durchsatz:

$$\lambda = \frac{\lambda_u + \lambda_o}{2}$$

  ➤ *Schritt* 2.2: Bestimmung von:

$$g(\lambda) = \sum_{i=1}^{N} f_i(\lambda \cdot e_i)$$

➤ Schritt 2.3: Abbruchbedingung:

$$\boldsymbol{g(\lambda) = K \pm \epsilon}$$

Berechne die Leistungsgrößen ausgehend von λ_i und ρ_i mit den Approximationsformeln.

Sonst:

If $\boldsymbol{g(\lambda) > K}$, set $\boldsymbol{\lambda_o = \lambda}$ and go back to Step **2.1**.
If $\boldsymbol{g(\lambda) < K}$, set $\boldsymbol{\lambda_u = \lambda}$ and go also back to Step **2.1**.

◆ Beispiel: Geschlossenes WS-Netz mit $N = 4$, $K = 3$, $\varepsilon = 0.001$ und:

i	e_i	$1/\mu_i$	m_i
1	1	0.5	2
2	0.5	0.6	1
3	0.5	0.8	1
4	1	1	∞

- ***Schritt*** 1: Initialisierung:

$$\lambda_u = \underline{0} \quad \text{und} \quad \lambda_o = \min_i \left\{ \frac{m_i \mu_i}{e_i} \right\} = \underline{2.5}$$

- ***Schritt*** 2:
  - ➤ *Schritt* 2.1: Durchsatz:

$$\lambda = \frac{\lambda_u + \lambda_o}{2} = \underline{1.25}$$

➤ Schritt 2.2: Berechnung der Funktionen $f_i(\lambda_i)$

$$\rho_1 = \frac{\lambda \cdot e_1}{\mu_1 \cdot m_1} = \underline{0.3125} \quad \text{and} \quad P_{m_1} = \underline{0.149}$$

$$f_1(\lambda_1) = 2\rho_1 + \rho_1 P_{m_1} = \underline{0.672}$$

$$f_2(\lambda_2) = \frac{\rho_2}{1 - \frac{2}{3}\rho_2} = \underline{0.5} \quad \text{with} \quad \rho_2 = \underline{0.375}\,,$$

$$f_3(\lambda_3) = \frac{\rho_3}{1 - \frac{2}{3}\rho_3} = \underline{0.75} \quad \text{with} \quad \rho_3 = \underline{0.5}\,,$$

$$f_4(\lambda_4) = \frac{\lambda_4}{\mu_4} = \underline{1.25}\,.$$

und damit:

$$g(\lambda) = \sum_{i=1}^{N} f_i(\lambda_i) = \underline{3.172}$$

➤ *Schritt* 2.3: Abbruchbedingung:

$$g(\lambda) > K \text{ therefore } \lambda_o = \lambda = 1.25$$

2. Iteration:

➤ Schritt 2.1: Durchsatz:

$$\lambda = \frac{\lambda_u + \lambda_o}{2} = \underline{0.625}$$

usw.

Die Tabelle zeigt den Iterationsverlauf für λ:

Step:	0	1	2	3	4	5	6	7	8	9	10
λ_l	0	0	0.625	0.9375	1.094	1.172	1.172	1.191	1.19 1	1.191	1.191
λ_u	2.5	1.25	1.25	1.25	1.25	1.25	1.211	1.211	1.20 1	1.196	1.194

Endgültiger Wert λ = 1.193 Exakter Wert: λ = 1.217

Exakte und Approximative Werte:

i	1	2	3	4
$\lambda_{i_{\mathrm{SUM}}}$	**1.193**	**0.596**	**0.596**	**1.193**
$\lambda_{i_{\mathrm{MVA}}}$	**1.218**	**0.609**	**0.609**	**1.218**
$\overline{K}_{i_{\mathrm{SUM}}}$	**0.637**	**0.470**	**0.700**	**1.193**
$\overline{K}_{i_{\mathrm{MVA}}}$	**0.624**	**0.473**	**0.686**	**1.217**

◆ Die Summationsmethode läßt sich auch auf WS-Netze mit mehreren Auftragsklassen erweitern.

Grenzwertanalyse:

Voraussetzungen:

- Es werden nur obere und untere Grenzen von Durchsatz und Antwortzeit berechnet:

$$\boldsymbol{\lambda}_{\mathbf{pes}} \leq \boldsymbol{\lambda} \leq \boldsymbol{\lambda}_{\mathbf{opt}} \quad \text{and} \quad \overline{\boldsymbol{T}}_{\mathbf{opt}} \leq \overline{\boldsymbol{T}} \leq \overline{\boldsymbol{T}}_{\mathbf{pes}}$$

- Nur für WS-Netze mit einer Auftragsklasse

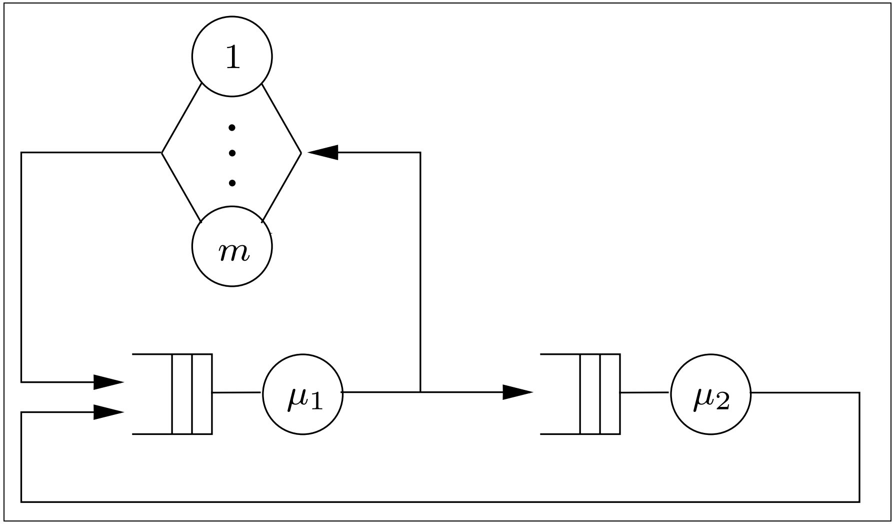
- Gilt nur für die folgenden Typen von WS-Netzen:
  - Typ A: Geschlossenes WS-Netz **ohne** Terminalknoten (IS-Knoten)
  - Typ B: Geschlossenes WS-Netz **mit** Terminalknoten (IS-Knoten)
  - Typ C: Offenes WS-Netz

◆ ABA - Asymptotic Bounds Analysis:

	Network Type	ABA Bounds
$\boldsymbol{\lambda}$	A	$\lambda(K) \le \min\left\{\frac{K}{x_{\text{sum}}}, \frac{1}{x_{\max}}\right\}$
	B	$\lambda(K) \le \min\left\{\frac{K}{x_{\text{sum}} + Z}, \frac{1}{x_{\max}}\right\}$
	C	$\lambda \le \frac{1}{x_{\max}}$
$\overline{T}$	A	$\overline{T}(K) \ge \max\left\{x_{\text{sum}}, K \cdot x_{\max}\right\}$
	B	$\overline{T}(K) \ge \max\left\{x_{\text{sum}}, K \cdot x_{\max} - Z\right\}$
	C	$\overline{T} \ge x_{\text{sum}}$

$$x_i = e_i / \mu_i \qquad x_{\max} = \max_i(x_i) \quad \text{and} \quad x_{\text{sum}} = \sum_i x_i$$

➤ Beispiel:



$$1/\mu_1 = 4.6\,, \quad 1/\mu_2 = 8\,, \qquad 1/\mu_3 = 120 = Z\,,$$
$$e_1 = 2\,, \qquad e_2 = e_3 = 1\,.$$

$$x_{\max} = \max\left\{\frac{e_1}{\mu_1}, \frac{e_2}{\mu_2}\right\} = \underline{9.2}\,, \quad x_{\mathrm{sum}} = \frac{e_1}{\mu_1} + \frac{e_2}{\mu_2} = \underline{17.2}\,,$$

$$Z = \frac{e_3}{\mu_3} = \underline{120}\,.$$

Throughput:

$$\lambda(K) \leq \min\left\{\frac{K}{x_{\mathrm{sum}} + Z}\,, \frac{1}{x_{\max}}\right\} = \min\left\{\frac{20}{137.2}\,; \frac{1}{9.2}\right\} = \underline{0.109}\,.$$

Mean response time:

$$\overline{T}(K) \geq \max\,\{x_{\mathrm{sum}}\,, K \cdot x_{\max} - Z\} = \max\,(17.2\,; 64) = \underline{64}\,.$$

Exakte Werte:

$$\lambda(K) = \underline{0.100} \text{ and } \overline{T}(K) = \underline{80.28}$$

◆ BJB - Balanced Job Bounds Analysis: $x_{ave} = x_{sum}/N$

	Network Type	BJB Bounds
λ	A	$\dfrac{K}{x_{sum} + (K-1)x_{max}} \le \lambda(K) \le \dfrac{K}{x_{sum} + (K-1)x_{ave}}$
	B	$\dfrac{K}{x_{sum} + Z + \frac{(K-1)x_{max}}{1+\frac{Z}{K\cdot x_{sum}}}} \le \lambda(K) \le \dfrac{K}{x_{sum} + Z + \frac{(K-1)x_{ave}}{1+\frac{Z}{x_{sum}}}}$
	C	$\lambda \le \dfrac{1}{x_{max}}$
$\overline{T}$	A	$x_{sum} + (K-1)x_{ave} \le \overline{T}(K) \le x_{sum} + (K-1)x_{max}$
	B	$x_{sum} + \dfrac{(K-1)x_{ave}}{1+\frac{Z}{x_{sum}}} \le \overline{T}(K) \le x_{sum} + \dfrac{(K-1)x_{max}}{1+\frac{Z}{K\cdot x_{sum}}}$
	C	$\dfrac{x_{sum}}{1-\lambda x_{ave}} \le \overline{T} \le \dfrac{x_{sum}}{1-\lambda x_{max}}$

➤ Beispiel:

- Durchsatz:

$$\underline{\mathbf{0.075}} \leq \boldsymbol{\lambda}(\boldsymbol{K}) \leq \underline{\mathbf{0.127}}$$

- Mittlere Antwortzeit:

$$\underline{\mathbf{37.70}} \leq \overline{\boldsymbol{T}}(\boldsymbol{K}) \leq \underline{\mathbf{146.8}}$$

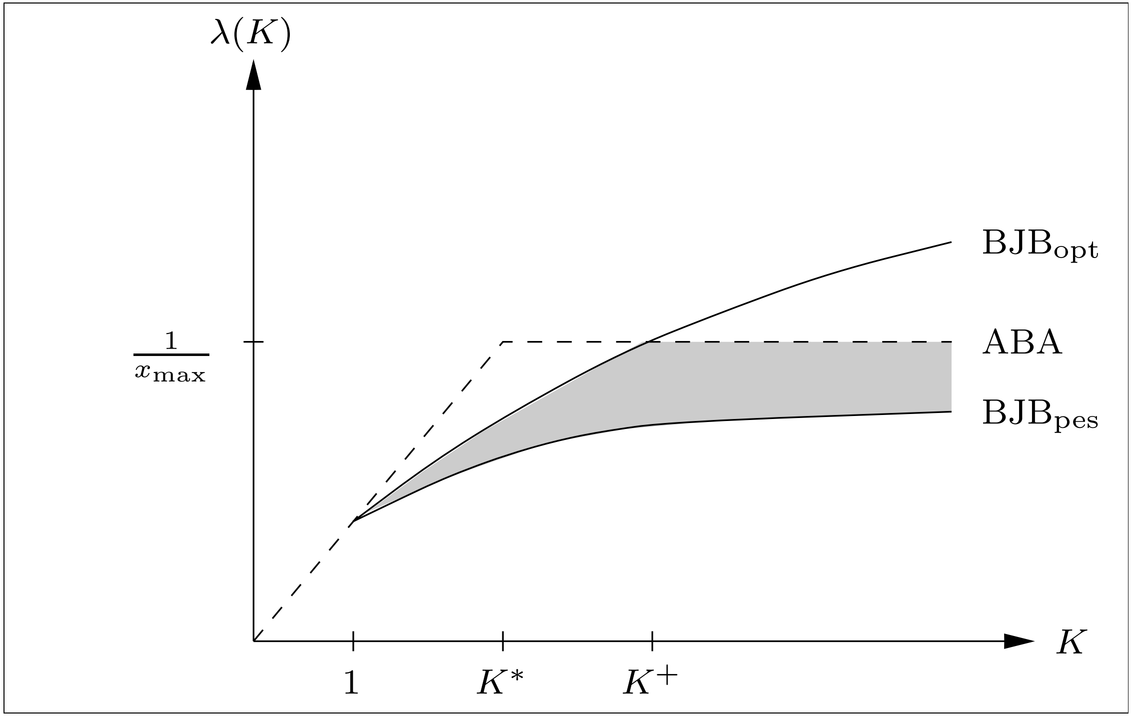
- Exakte Werte:

$$\boldsymbol{\lambda}(\boldsymbol{K}) = \underline{\mathbf{0.100}} \text{ and } \overline{\boldsymbol{T}}(\boldsymbol{K}) = \underline{\mathbf{80.28}}$$

- ABA Werte:

$$\boldsymbol{\lambda}(\boldsymbol{K}) = \underline{\mathbf{0.109}} \text{ and } \overline{\boldsymbol{T}}(\boldsymbol{K}) = \underline{\mathbf{64}}$$

- Durchsatz in Abhängigkeit von K:



- Mittlere Antwortzeit in Abhängigkeit von K:

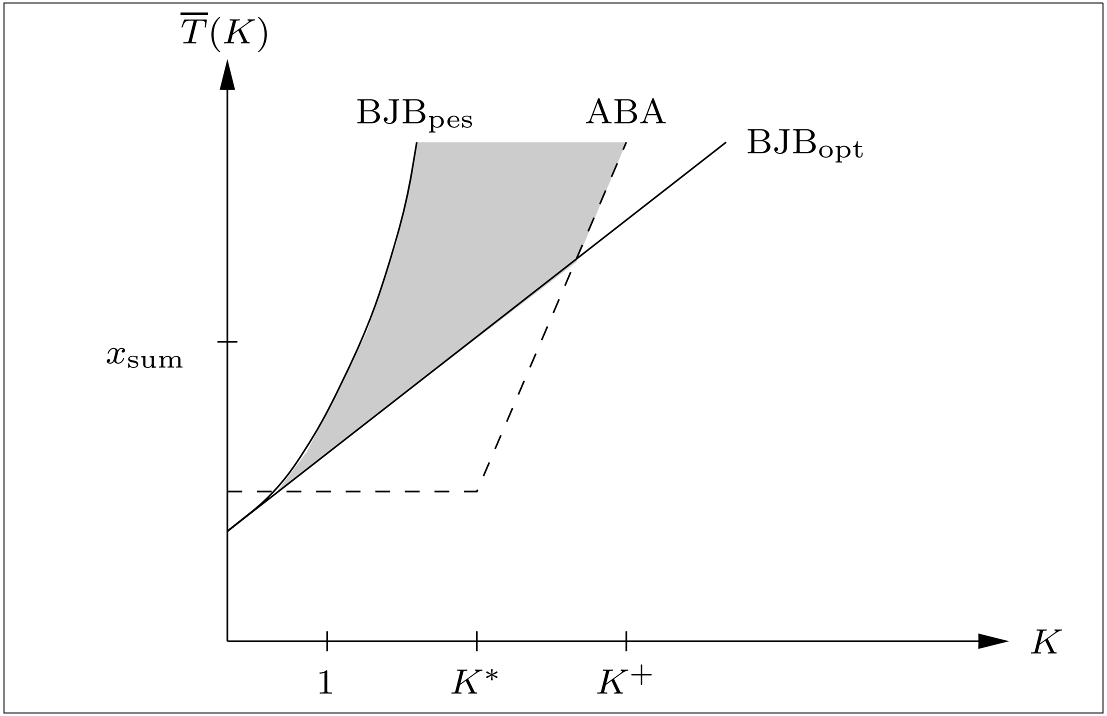


Table 0.1 Comparison of the approximation algorithms for product-form queueing networks.

Algorithms	Advantages	Disadvantages
Bard-Schweitzer (BS)	Very low storage and time requirement	No multiple server nodes Low accuracy
SCAT	Good accuracy Very low storage requirement compared with MVA or convolution	Needs more iterations than BS
SUM	Easy to understand and implement Low storage and time requirement Easy to extend to non-product-form networks	Accuracy is not very high (but sufficient for most applications)
ABA, BJB	Well suited for a bottleneck analysis In the design phase, to obtain a rough prediction (insight, understanding) of the performance of a system Extremely low storage and time requirement	Only for single class networks Only upper and lower bounds