A Distributed QoS Scheduler for Smoothing Output Traffic of Input Buffered Switches

Man-Ting Choy and Tony T. Lee

Department of Information Engineering,
The Chinese University of Hong Kong
{mtchoy1, ttlee}@ie.cuhk.edu.hk
http://bblab.ie.cuhk.edu.hk/index.html [*]

Abstract. To provide stringent service guarantees such as latency and backlog bounds for input-buffered switches, a set of scheduling algorithm and admission control strategy is proposed. This set of traffic control strategy is primarily based on a single-server scheduling algorithm called Smoothed Round Robin (SRR). SRR possesses a number of advantages which are very attractive to the implementation of input buffered switch. SRR is on order $O(1)$ which requires minimal computational complexity. Secondly, SRR gives good delay bounds and fairness performance for each session. Thirdly, SRR can decompose a sequence into fixed size groups. In this way, by maintaining a SRR scheduler in each output port, scheduling can be performed in a distributed manner which largely reduces the complexity of the algorithm.

1 Introduction

The development of multirate interconnection networks comes from the necessity of developing a new generation of switches for broadband services which require stringent Quality of Services (QoS) guarantees, such as end-to-end delay, jitter and minimum bandwidth requirements. The nonblocking conditions for multirate traffic with different types of networks and comparisons on their complexity are established in [1]. Related researches [2] [3] [4] [5] [6] [7] have been carried out but these studies do not give us a complete set of traffic scheduling and routing algorithm to guarantee the QoS requirement.

A novel routing scheme for large-scale packet switches called path switching was proposed in [8]. This scheme provides end-to-end QoS guarantees in Clos network. Path switching is a compromise of static routing and dynamic routing schemes. The basic idea of this scheme is to use a set of predetermined connection patterns of central switching modules, and these connection patterns are used repeatedly in a periodical manner. The capacity requirement on each session can be satisfied in the long run, and the computation of route assignment on-the-fly can be avoided.

[*] The work described in this paper was substantially supported by grants from the Research Grants Council of the Hong Kong Special Administrative Region. (Project no. CUHK4380/02E and Direct Grant 2005/06 2050360).

However, the output traffic of this scheme may become bursty. In path switching, a token is considered as a middle module in a particular time slot through which packets can transverse from a particular input module to a particular output module. In this case, tokens are assigned to each input module to satisfy all their capacity requirement. While this assignment problem can be solved by edge-coloring of bipartite graph, this easy solution cannot guarantee an uniform distribution of tokens, which is necessary to achieving smooth output traffic and tight delay bounds for each session.

Recently, the traffic matrix decomposition approach of path switching [8] was adopted by Chang et al [9] [10], in which a scheduling algorithm for QoS guarantees of input queued switches was developed. Their algorithm consists of two parts: an offline part that breaks down the rate matrix into a set of permutation matrices, and an online part that schedules these matrices using Weighted Fair Queueing (WFQ). However, the time complexity of this algorithm is quite large (the offline part is of $O(N^{4.5})$ and the online part is of $O(logN)$ for a $N \times N$ switch). The number of permutation matrices that results from this decomposition is in the order of N^2. Moreover, the worst case delay can be very large since the decomposition is done randomly and the resulting permutation matrices would not be able to provide smooth output traffic for every session. The load balanced approach in [10], although is much simpler, would give out-of-order packets problem. Therefore, this algorithm is not quite practical for large scale packet switch.

In this paper, a set of scheduling algorithm and admission control strategy is provided in order to guarantee smoother output traffic while maintaining low operational complexity. This set of traffic control algorithm is based on a fair scheduling algorithm called Smoothed Round Robin (SRR) [11]. In Section 2, the basic concept of SRR will be explained. In Section 3, we will explain the methodology of implementing SRR in input-buffered switch and also the admission control strategy needed. In Section 4, the performance of the scheduler will be discussed. By applying network calculus, the deterministic QoS guarantees are derived in Section 5. At last, we will conclude our work in Section 6.

2 The Smoothed Round Robin

Smoothed Round Robin is a simple scheduling algorithm which has the major advantage of its O(1) time computational complexity. Two key data structures of the scheduler are the Weight Spread Sequence (WSS) and the Weight Matrix (WM). The WSSs are defined as follows:

1) The first WSS $S^1 = 1$.

2) The kth WSS is

$$S^k = S^{k-1}, k, S^{k-1} \tag{1}$$

where $k > 1$ and $1 \leq i \leq 2^k - 1$.

For the Weight Matrix, each flow is assigned with a weight in proportion to its reserved rate and the set of weights is assumed to be {1, 2, 3, ..., 2^k -1}. Then the weight of $flow_f$ can be coded in binary as

$$w_f = \sum_{n=0}^{k-1} a_{f,n} 2^n, \text{where } a_{f,n} = \{0, 1\}.$$

The Weight Vector of $flow_f$ is defined as

$$WV_f = \{a_{f,(k-1)}, a_{f,(k-2)}, ..., a_{f,0}\}. \tag{2}$$

Then the Weight Matrix is defined as

$$WM = \begin{bmatrix} WV_1 \\ WV_2 \\ WV_3 \\ \vdots \\ WV_N \end{bmatrix} \tag{3}$$

for N input flows. The columns of the Weight Matrix are named as $column_{k-1}$, $column_{k-2}$, $\ldots$, $column_0$ from left to right respectively. Notice that k is the number of columns in the WM. To schedule packets, SRR scans the WSS sequence term by term. When the value of the term is i, the $column_{k-i}$ of the WM is chosen. In this column, the scheduler will scan the terms from top to bottom. When the term is not 0, the scheduler will serve the corresponding flow.

For example, given there are three flows with weights $w_a = 2$, $w_b = 3$, $w_c = 5$, the Weight Matrix and Weighted Spread Sequence are

$$WM = \begin{bmatrix} WV_a \\ WV_b \\ WV_c \end{bmatrix} = \begin{bmatrix} 0\,1\,0 \\ 0\,1\,1 \\ 1\,0\,1 \end{bmatrix} \text{ and } WSS = 1, 2, 1, 3, 1, 2, 1$$

In this way, the first column would be served first, which only contains flow C. Then the second column get served, which contains flows A and B. The overall result, CABCBCCABC, would be generated when the whole WSS was processed.

3 Scheduling in Input Buffered Switch by Applying the Concept of SRR

The concept of path switching is adopted here, where the time axis is divided into frames of time-slots and tokens (port-to-port path) are assigned to input flows to fulfill their capacity requirements. To provide input buffered switch with smooth output traffic and deterministic QoS guarantees, tokens have to be assigned uniformly. This can be achieved by incorporating SRR into input buffered switch. Notice that SRR is originally designed to support variable size packets by means of deficit counter. It can be easily adopted in the switching environment which requires fixed-sized packets.

3.1 Admission Control

To begin with, a set of traffic admission control has to be set up. A weight matrix is maintained at each input/output port such that the weight of an incoming flow is recorded in the WMs of input port as well as its destined output port. However, the column sum of these WMs are predefined such that an incoming request is rejected if the inclusion of its weight into the WMs would violate the column sum restriction, either at the input or the output port. All the WMs are having the same set of column sum restriction.

3.2 Token Assignment

To assign tokens, SRR would be performed in each output port. However, tokens cannot be assigned simply according to the result of SRR since different output port may reserve token for the same input port but each input port can only process one packet. In this way, permutation matrix cannot be formed in that particular time-slot. This is the reason why token assignment algorithm has to be centralized as it has to cooperate with both the input and output port. However, with the restrictions in the WMs, we can have an easier solution. The restriction in the output port has allowed a simple partitioning of tokens into groups and the restriction in the input port has allowed a simple small-scale rescheduling within each group to form the permutation matrices.

For example, given the following capacity requirement matrix

$$R = \begin{bmatrix} 7 & 5 & 2 & 2 \\ 0 & 6 & 7 & 3 \\ 3 & 3 & 6 & 4 \\ 6 & 2 & 1 & 7 \end{bmatrix},$$

which tells us the weight requirement from each input port to each output port, the Weight Matrices for output ports (columns of R) are

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Notice that the column sums of these WMs are identical (2, 3, 2). This satisfies the output port restriction. On the other hand, for the input ports (rows in R), the WMs are

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Again, the column sums here are also identical. This satisfies the input port restriction. By performing SRR at each output port of the switch, we have

A D	A C D	A D	A C	A D	A C D	A D
A B	B C D	A B	A C	A B	B C D	A B
B C	A B C	B C	B D	B C	A B C	B C
C D	A B D	C D	B D	C D	A B D	C D

where A, B, C and D represent the tokens for the four input ports. Notice that permutation matrices cannot be formed in each time-slot. However the tokens are partitioned in the same format and by shuffling the tokens inside each partition (for example, by edge coloring of bipartitle graph [8]), we have

A D	A C D	A D	A C	A D	A C D	A D
B A	C D B	B A	C A	B A	C D B	B A
C B	B A C	C B	B D	C B	B A C	C B
D C	D B A	D C	D B	D C	D B A	D C

In this way, the tokens are distributed uniformly and thus output traffic is smoother. Since the number of elements in each group should be small and the rescheduling within group can be done in parallel fashion, complexity of this algorithm should also be small.

4 Performance Analysis

In this section, we would first discuss the relative fairness of the SRR, which is essential in obtaining deterministic QoS guarantees, which will be discussed in the next section. The scheduling delay bound of the proposed scheduler would also be discussed here.

4.1 Relative Fairness of Smoothed Round Robin

To analyze the fairness of scheduling algorithm, Golestani [12] proposed to find the maximum difference between the normalized service received by two backlogged flows over any time interval as a fairness index, which can be expressed as

$$RF = \max\left(\left|\frac{V_f(\tau,t)}{w_f} - \frac{V_g(\tau,t)}{w_g}\right|\right)$$

where $V(\tau,t)$ represents the amount of service received by a session in any time interval τ to t and w as the weight of that session. Given k is the order of the current WSS used by SRR, the author has showed in [11] that

$$\left|\frac{V_f(0,t)}{w_f} - \frac{V_g(0,t)}{w_g}\right| \leq \frac{k}{2\min(w_f,w_g)}, \tag{4}$$

which does not represent the real relative fairness index. This is because in (4), it is assumed that the backlog would always start from the beginning of WSS, which is not true. For example, when $w_f = 3$ and $w_g = 2$, the output is F, G, F, F, G and

$$\frac{V_f(0,t)}{w_f} - \frac{V_g(0,t)}{w_g} = \left\{\frac{1}{3}, -\frac{1}{6}, \frac{1}{6}, \frac{1}{2}, 0\right\}$$

for t={1, 2, 3, 4, 5}. If the backlog start at $t = 2$, the output sequence becomes F, F, G, F, G and then

$$\frac{V_f(2,t)}{w_f} - \frac{V_g(2,t)}{w_g} = \left\{\frac{1}{3}, \frac{2}{3}, \frac{1}{6}, \frac{1}{2}, 0\right\}$$

for $t = \{3, 4, 5, 6, 7\}$. In this case, the value $\frac{2}{3}$ is larger than the bound $\frac{1}{2}$ in (4).

While we cannot obtain the value of RF directly from (4), we are not far from the solution. As shown in Fig. 1, the value of RF is the sum of LRF and SRF, which represent the larger and smaller parts of RF away from the x-axis respectively. Now, as LRF is actually given in (4), we only have to find the value of SRF, which turns out to be having a smaller bound than LRF.

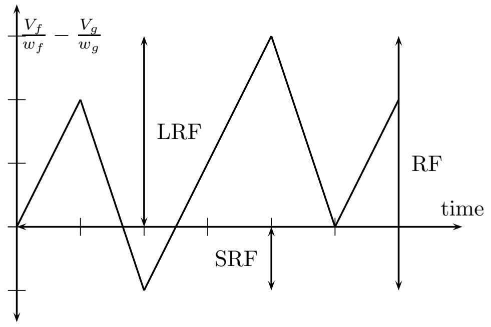

Fig. 1. Relative Fairness - Definition of LRF and SRF

Theorem 1. *For any pair of backlogged flows f and g in SRR, at any time instance t, we have*

$$SRF \leq \frac{(k-1)\max(w_f, w_g) + 2}{2w_f w_g} \tag{5}$$

where k is the order of the current WSS used by SRR.

Proof. The proof is shown in the Appendix. □

Theorem 2. *For any pair of backlogged flows f and g in SRR, where the backlog started at time τ and at any time instance t, we have the relative fairness index*

$$RF = \max\left(\left|\frac{V_f(\tau,t)}{w_f} - \frac{V_g(\tau,t)}{w_g}\right|\right) \leq \frac{(2k-1)\max(w_f, w_g) + 2}{2w_f w_g} \tag{6}$$

where k is the order of the current WSS used by SRR.

Proof.

$$\begin{aligned} RF &\le LRF + SRF \\ &\le \frac{k}{2\min(w_f, w_g)} + \frac{(k-1)\max(w_f, w_g) + 2}{2w_f w_g} \\ &= \frac{(2k-1)\max(w_f, w_g) + 2}{2w_f w_g} \end{aligned}$$

Therefore Theorem 2 is proved. □

4.2 Delay Bound of the Proposing Scheduling Algorithm

In this section, we will show the single packet delay bound of this scheduler, which represents the time for a head of line packet to be completely transmitted over the switch. This bound is valid without any input traffic constraint, such as traffic envelope.

Theorem 3. *Suppose the weight assigned to $flow_f$ is w_f, the delay encountered by this flow in the input buffered switch using SRR is bounded by*

$$d_{SRR} \le \frac{2(w_f + w_G)}{w_f} + 2N_b \tag{7}$$

where w_G is the weight of flows other than $flow_f$ and N_b is the maximum column sum in WM.

Proof. A flow is visited when one of its coefficients is visited by SRR. Therefore, the delay bound of a flow is the maximum value of the intervals between two adjacent visits by SRR. By assuming $2^i \le w_f \le 2^{i+1} - 1$, the chain with the maximum length between two adjacent occurrences of element $(k-i)$ is $S^{k-i-1}, (k-y), S^{k-i-1}$ for $y < i$ and $a_{f,y} = 0$. In [11], the author has shown that the delay (V_{cnt}) as mapped by $S^{k-i-1}, (k-y), S^{k-i-1}$ and $column_i$ is

$$V_{cnt} \le \frac{1}{2^i}\left(w_f + w_G - \sum_{n=0}^{i-1} 2^n \sum_{m=1}^{N} a_{m,n}\right) + \sum_{m=1}^{N} a_{m,y} \tag{8}$$

When applied in a switch, the extra delay resulted is the number of elements in $column_i$, thus

$$\begin{aligned} d_{SRR} &\le \frac{1}{2^i}\left(w_f + w_G - \sum_{n=0}^{i-1} 2^n \sum_{m=1}^{N} a_{m,n}\right) + \sum_{m=1}^{N} a_{m,y} + \sum_{m=1}^{N} a_{m,i} \qquad (9) \\ &\le \frac{1}{2^i}\left(w_f + w_G\right) + \sum_{m=1}^{N} a_{m,y} + \sum_{m=1}^{N} a_{m,i} \qquad (10) \\ &\le \frac{2\left(w_f + w_G\right)}{w_f} + N_b + N_b \qquad (11) \end{aligned}$$

Hence, Theorem 3 is proved. □

5 Deterministic QoS Guarantees

A model to study the deterministic QoS guarantees for each session is developed. Using this model, upper bounds on delay and backlog can be established, assuming each input traffic stream is under leaky-bucket rate control and there is no packet loss due to buffer overflow and delay bound violation.

5.1 Network Calculus

To model the service received by each session, we can use the concept of service curve [13], which represents the least amount of service provided by the network element to a data session during its busy period. Fig. 2 shows an arrival curve $A()$ together with a service curve $S()$. The service curve is a straight line with the slope representing the service rate reserved for a particular session at the network element. In this way, the delay and backlog bounds can be easily calculated as shown in the figure.

As discussed in our previous work [14], given an arrival curve with burstiness constraint (σ, ρ, C), where σ is the bucket size, ρ is the arrival rate and C is the maximum rate of tokens flowing out from the bucket, the delay of a packet is bounded by

$$D = v + \frac{\sigma}{C-\rho}\left(\frac{C}{g-1}\right) \tag{12}$$

while the backlog bound is

$$B = \begin{cases} vg + \frac{C-g}{C-\rho}\sigma & \text{if } v < \frac{\sigma}{C-\rho} \text{ and } C > g \\ vC & \text{if } v < \frac{\sigma}{C-\rho} \text{ and } C \leq g \\ v\rho + \sigma & \text{otherwise} \end{cases} \tag{13}$$

where g denotes the reserved rate of the session and v is a parameter related to the service curve as shown in Fig. 2 and will be discussed in the next subsection.

5.2 Obtaining the Service Curve of the Proposing Scheduling Algorithm

As shown in Fig. 3, by drawing all the possible schedulers output on the same graph, a service curve, which is the lower bound of all the distributions can be obtained. Denotes w_f as the weight of the session we are interested in and w_G as the aggregated weights of other sessions sharing the same output port. Denotes $V_f(\tau, t)$ as the packets served for the session of interest from the start of backlog τ to time t and $V_G(\tau, t)$ as the packets served for other sessions, then the value of v, which denotes the minimum delay needed to guarantee a steady service of reserved rate from the switch, can be obtained. According to Fig. 3, we have

$$\frac{V_f(\tau, t)}{V_f(\tau, t) + V_G(\tau, t) - v} \geq \frac{w_f}{w_f + w_G} \tag{14}$$

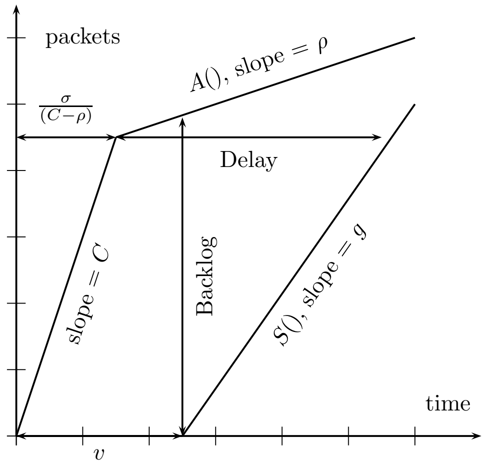

Fig. 2. Arrival and Service Curves

which implies that

$$v \geq \frac{V_G(\tau,t)w_f - V_f(\tau,t)w_G}{w_f} \tag{15}$$

Therefore, the smallest value of v can be obtained once we can find the maximum value of $V_G(\tau,t)w_f - V_f(\tau,t)w_G$.

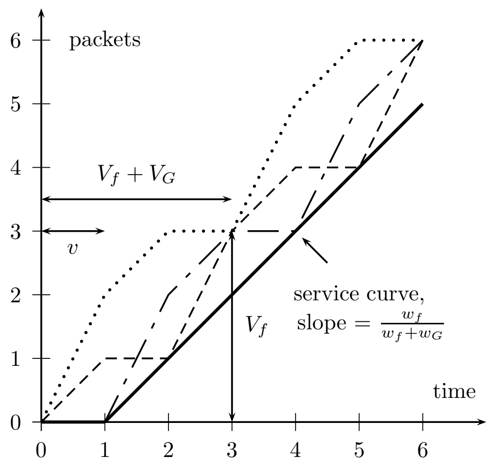

Fig. 3. Resultant Service Curve

Theorem 4. *Given w_f as the weight of the session we are interested in and w_{g_i} as the weights of other flows destined to the same output port as $flow_f$, v*

is bounded by

$$v \geq \sum_i \frac{(2k_i + 1)\max(w_f, w_{g_i}) + 2}{2w_f}$$

where k_i are the smallest integers that $2^{k_i} \geq \max(w_f, w_{g_i})$.

Proof.

$$|V_G w_f - V_f w_G| \leq \sum_i \left[|V_{g_i} w_f - V_f w_{g_i}| + \max(w_f, w_{g_i})\right] \tag{16}$$

$$\leq \sum_i \left[\frac{(2k_i - 1)\max(w_f, w_{g_i}) + 2}{2} + \max(w_f, w_{g_i})\right] \tag{17}$$

$$\leq \sum_i \frac{(2k_i + 1)\max(w_f, w_{g_i}) + 2}{2} \tag{18}$$

The last term of (16) represents the extra delay resulted from the rescheduling within each group. (6) was applied to give (17). Then from (15), we have

$$v \geq \frac{V_G(\tau, t) w_f - V_f(\tau, t) w_G}{w_f} \tag{19}$$

$$\geq \sum_i \frac{(2k_i + 1)\max(w_f, w_{g_i}) + 2}{2w_f} \tag{20}$$

Thus, Theorem 4 is proved. □

6 Conclusion

In this paper, we applied smoothed round robin to input buffered switches and yielded desired result. SRR has short delay bounds and good fairness performance, thus the application of SRR in input buffered switch will allow a less-bursty output traffic. We have derived the relative fairness of SRR and the deterministic QoS guarantees of the proposed scheduling algorithm by using the concept of Network Calculus.

References

1. Melen, R., Turner, J.S.: Nonblocking networks for fast packet switching. In: IEEE Infocom '89. Volume 2. (1989) 548–557
2. Li, S., Ansari, N.: Input-queued switching with QoS guarantees. In: IEEE Infocom'99. (1999) 1152–1159
3. Hung, A., Kesidis, G., Mckeown, N.: ATM input-buffered switches with guaranteed-rate property. In: IEEE ISCC'98. (1998) 331–335
4. Liotopoulos, F., Chalasani, S.: Semi-rearrangeably nonblocking operation of Clos networks in the multirate environment. IEEE Transactions on Networking. **4** (1996) 281–291

5. Naraghi-Pour, M., Hegde, M., Suresh, S.: Scheduling multi-rate traffic in a time-multiplex switch. In: Proceedings on Information Theory'94. (1994) 406
6. Valdimarsson, E.: Blocking in multirate interconnection networks. IEEE Transactions on Networking. **42** (1994) 2028–2035
7. Favalli, L.: Rearrangeability conditions for multirate Benes networks. In: GLOBECOM' 93. (1993) 734–738
8. Lee, T., Lam, C.: Path switching: A quasi-static routing scheme for large-scale ATM packet switches. IEEE Journal on Selected Areas in Communications **15**(5) (1997) 914–924
9. Chang, C.S., Chen, W.J., Huang, H.Y.: Birkhoff-von Neumann input-buffered crossbar switches for guaranteed-rate services. IEEE Trans. Commun. **49** (2001) 1145–1147
10. Chang, C.S., Chen, W.J., Huang, H.Y.: Providing guaranteed rate services in the load balanced Birkhoff-von Neumann switches. In: IEEE Infocom 03. Volume 3. (2003) 1622–1632
11. Guo, C.: SRR: An O(1) time-complexity packet scheduler for flows in multiservice packet networks. IEEE Transactions on Networking. **12**(6) (2004) 1144–1155
12. Golestani, S.: A self-clocked fair queueing scheme for broadband applications. In: IEEE Infocom 94. (1994) 636–646
13. Cruz, R.L.: Quality of service guarantees in virtual circuit switched networks. IEEE Journal on Selected Areas in Communications **13** (1995) 1048–1056
14. Chan, M.C., To, P., Lee, T.T.: Per-connection performance guarantees for cross-path ATM packet switch. In: ATM Workshop 1999. (1999) 469–474

Appendix: Proof of Theorem 1

For easier presentation, we multiply both sides of (5) with $w_f w_g$, then we define

$$WSRF = (w_f w_g)SRF \leq \frac{k-1}{2}\max(w_f, w_g) + 1$$

This theorem is proved by induction as follows

1) It is true for $k = 1$ and 2,

2) Suppose that the inequality is correct using a kth WSS, i.e., for any pair of w_f and w_g, we have

$$WSRF \leq \frac{k-1}{2}\max(w_f, w_g) + 1$$

Then for any pairs of f' and g' using a $(k+1)$th WSS, $w_{f'}$ and $w_{g'}$ can be expressed as

$$\begin{aligned} &w_{f'} = 2w_f + a_{f',0}, w_{g'} = 2w_g + a_{g',0} \\ &\qquad\text{where } w_{f'} > 1, w_{g'} > 1, a_{f',0}, a_{g',0} = 1 \text{ or } 0. \end{aligned}$$

Therefore, the service sequence of flow f' and g' can be expressed as

$$S^{k+1}(f', g') = S^k(f, g), \{a_{f',0}.f, a_{g',0}.g\}, S^k(f, g).$$

Here we just show the last case where $w_{f'} = 2w_f+1$ and $w_{g'} = 2w_g + 1$. When $V_{f'} = V_f$ and $V_{g'} = V_g$,

$$\begin{aligned}
WSRF &= |(2w_f+1)V_g - (2w_g+1)V_f| \\
&\le |2w_fV_g - 2w_gV_f| + |V_g - V_f| \\
&\le [(k-1)\max(w_f,w_g)+2] + [\max(w_f,w_g)] \\
&\le k\max(w_f,w_g)+2 \\
&\le \frac{k}{2}\max(2w_f,2w_g) + \frac{k}{2} + 1 && (\text{for } k \ge 2) \\
&\le \frac{k}{2}\max(w_{f'},w_{g'}) + 1 && (w_{f'} = 2w_f+1)
\end{aligned}$$

When $V_{f'} = w_f + 1$ and $V_{g'} = w_g$,

$$\begin{aligned}
WSRF &= |(2w_f+1)w_g - (2w_g+1)(w_f+1)| \\
&= |w_g + w_f + 1| \\
&\le 2\max(w_f,w_g)+1 \\
&\le \frac{k}{2}\max(w_{f'},w_{g'}) + 1 && (\text{for } k \ge 2)
\end{aligned}$$

When $V_{f'} = w_f + 1$ and $V_{g'} = w_g + 1$,

$$\begin{aligned}
WSRF &= |(2w_f+1)(w_g+1) - (2w_g+1)(w_f+1)| \\
&= |w_g - w_f| \\
&\le \frac{k}{2}\max(w_{f'},w_{g'}) + 1
\end{aligned}$$

When $V_{f'} = w_f + 1 + V_f$ and $V_{g'} = w_g + 1 + V_g$,

$$\begin{aligned}
WSRF &= |(2w_f+1)(w_g+1+V_g) - (2w_g+1)(w_f+1+V_f)| \\
&\le |2w_gV_f - 2w_fV_g| + |w_g - V_g - w_f + V_f| \\
&\le [(k-1)\max(w_f,w_g)+2] + [\max(w_f,w_g)] && (\text{since } w_f \ge V_f) \\
&\le k\max(w_f,w_g)+2 \\
&\le \frac{k}{2}\max(w_{f'},w_{g'}) + 1
\end{aligned}$$

Hence, for different combinations of $V_{f'}$ and $V_{g'}$ in the last case, we have

$$\begin{aligned}
WSRF &\le \frac{k}{2}\max(w_{f'},w_{g'}) + 1 \\
SRF &\le \frac{k\max(w_{f'},w_{g'})+2}{2w_{f'}w_{g'}}
\end{aligned}$$

Therefore, Theorem 1 follows by induction. □