Application of the Goore Scheme to Turbulence Control for Drag Reduction (I)
- Improvement of the Goore Scheme -

Changhoon Lee*
School of Electrical and Mechanical Engineering, Yonsei University, Seoul 120-749, Korea
Namhyun Kim
Department of Applied Mathematics, Hongik University, Seoul 121-791, Korea
John Kim
Department of Mechanical and Aerospace Engineering, UCLA, USA

We investigate the possibility of application of the Goore Scheme to turbulence control for drag reduction. In Part I, we examine the performance of the original Goore Scheme by applying it to a simple one-dimensional problem. For the application of the scheme to turbulence control, we extend the scheme's capability so that it can treat multi-dimensional problems and examine its validity theoretically. The convergence of the extended scheme with a dynamic memory is faster by an order of magnitude than the original scheme. In Part II, we apply the proposed scheme to reduce drag for turbulent channel flows through direct numerical simulation.

Key Words : Turbulence Control, Goore Scheme, Direct Numerical Simulation

1. Introduction

Near-wall streamwise vortices have been known to be responsible for the skin friction caused by turbulence on a flat plate. For the last decade, many researchers have tried to manipulate these structures by applying appropriate control inputs to reduce drag on the flat plate. Several were very successful and they reduced drag by more than 20% in their numerical simulations at low Reynolds number (Choi *et al.* 1994, Bewley and Moin 1995, Lee *et al.* 1997, Lee *et al.* 1998, Koumoutsakos 1999). Their strategies are based on physical intuition, control theories such as the optimal or suboptimal control theory, or nonlinear neural networks. The performance of these controls, however, depends on the choice of control parameters such as the cost function. Whenever a control scheme utilizing one of the above schemes is tested to obtain drag reduction, a great effort to find optimum control parameters should be made to achieve the goal. Some control schemes with wrongfully chosen parameters do not achieve drag reduction at all. This undesirable performance is mainly due to the combination of the nonlinear nature of turbulence and the complexity associated with control schemes.

On the other hand, a more systematic approach using linear control theory has been adopted to delay transition in a developing turbulent boundary layer (Joshi et al. 1997, Bewley and Liu 1998). This linear control was later applied to a fully developed turbulent channel flow with a partial success (Cortelezzi et al. 1998, Lee et al. 2001). Because of its own limitation, the linear control schemes are not expected to perform well in fully developed turbulence.

* Corresponding Author,
E-mail : clee@yonsei.ac.kr
TEL : +82-2-2123-2846; **FAX** : +82-2-312-2159
Division of Mechanical Engineering, School of Electrical and Mechanical Engineering, Yonsei University, Shinchon-dong 134, Seodaemun-ku, Seoul 120-749, Korea. (Manuscript **Received** April 30, 2001; **Revised** August 30, 2001)

To control turbulence, which is a strongly nonlinear dynamical system with an almost infinite number of degrees of freedom, we need a robust nonliner control scheme that can be applied to any nonlinear dynamical system even with unknown mechanism. We find the Goore Scheme to be a good candidate for such application. The Goore Scheme is a reinforcement learning scheme that can be easily applied to any nonlinear problem. The main advantage of this scheme is that it does not require any specific knowledge on how the system works. Instead, only the control input and the measurable output are necessary for the scheme to perform properly. Recently, Tung and Kleinrock (1995) provided a theoretical background for this scheme. We investigate the feasibility of the scheme for turbulence control.

In Part I, we examine the original Goore Scheme, and in Sec 2, we investigate its performance. We extend the scheme for multi-dimensional problems and validate it mathematically in Sec. 3. We also accelerate the convergence of the scheme by introducing a dynamically varying memory variable in Sec. 4 and present conclusions in Sec. 5. Therefore, the Part I will serve as a good tutorial for the improved Goore Scheme for application to any control not restricted to turbulence control. In Part II (Lee and Kim, 2001), we apply the proposed Goore Scheme to the control of a turbulent channel flow for drag reduction.

2. The Goore Scheme

Originally the Goore Scheme was developed to control distributed systems. The scheme can be best explained by the Goore Game of Tsetlin (1964). Suppose there is a group consisting of N players and a referee. Each time all players vote *yes* or *no* with the referee counting the number of *yes* votes. Depending on the reward probability function $r(f)$ which is a function of the percentage of the *yes* vote, f, the referee rewards each player with probability of r or penalizes with probability of $1-r$ no matter what the player voted. Then each player makes a decision for the

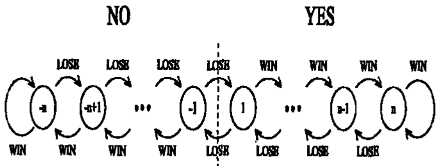

Fig. 1 Design of an automaton

next vote based on the reward or penalty. Here each player is isolated from each other and even does not know the distribution of the reward probability function. If there exists a value of percentage, f^*, for which the reward probability function has its maximum value, the Goore Scheme can show the following. Regardless of the number of the players, it is always possible to construct a decision mechanism of each player such that f^* percentage of players always votes *yes* after sufficient number of trials. Such a decision mechanism, also known as an automaton, is shown in Fig. 1. Each player can have a memory value ranging from $-n$ to n except zero. If the memory value is positive/negative, the player votes *yes*/*no*. When a player is rewarded/penalized, the position of the memory of the player moves away from/toward zero. When the memory hits n, it stays there until the player gets penalized. When the memory value is 1 or -1, a penalty can flip the sign, and only when this happens, the player flips its vote from *yes* to *no* or from *no* to *yes* depending on the past value of his memory. This means that even when a player gets a penalty, the player sticks to its previous vote until he gets more penalties, enough to flip the sign of his memory.

For example, when k out of N players vote *yes*, the reward probability function r is a function of k/N. Each player gets rewarded or penalized with a probability of r or $1-r$. An example of a reward function $r(k/N)$, with its maximum at $k^*/N=0.3$, is shown in Fig. 2. With the automaton of Fig. 1, after enough trials, the probability that 30 percent of players always vote *yes* approaches 1 when the maximum allowable memory, n, is large enough. Such a probability distribution was theoretically estimated by Tung and Kleinrock (1995). We briefly explain it here.

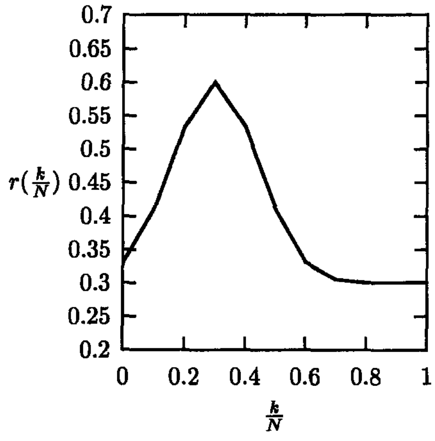

Fig. 2 One-dimensional reward function $r(k/N)$. $k^*/N=0.3$ and $N=10$

Suppose that k out of N players vote *yes* and $N-k$ players vote *no*. k players who vote *yes* have positive memory values ranging from 1 to n, and the other $N-k$ players have negative memory values ranging from $-n$ to 1. Only when the memory value of a player changes from 1 to -1 or from -1 to 1, however, the number of *yes* votes can change. Otherwise, the number of *yes* votes remains the same. Therefore, consider a case in which the number of *yes* votes changes. The probability that the number of *yes* votes decreases by one becomes k/N and the probability otherwise is $(N-k)/N$. Such a process can be modeled as a Markov chain (Karlin & Taylor, 1975). Hence the equilibrium probability that k out of N players votes *yes* becomes

$$\Pi(k)=2^{-N}\binom{N}{k} \qquad (1)$$

which means that, as N increases, the distribution becomes normal and has a maximum value at $k=N/2$. We have not considered how much time each state spends. If the time spent at each state is $\tau_n(k)$, the probability that the system stays at k-state becomes

$$\Phi(k)=\frac{\Pi(k)\,\tau_n(k)}{\sum_{k'=0}^{N}\Pi(k')\,\tau_n(k')} \qquad (2)$$

The persistence time, $\tau_n(k)$ can be estimated by the method of Tung and Kleinrock (1995) as

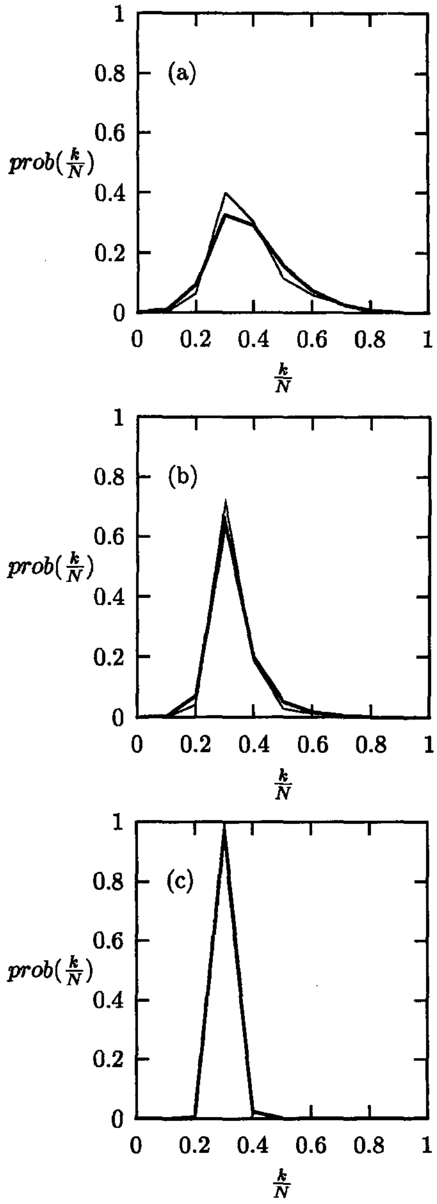

Fig. 3 Probability density function of the *yes* percentage in one-dimensional problem for 3 different memory values: (a) $n=5$; (b) $n=10$ (c) $n=20$. The thick solid line denotes the numerically tested result and the thin solid line denotes the theoretical estimate. The number of entities N is 10.

$$\tau_n(k)=\tau_n(r(k/N))=\frac{1}{1-2r}\left(1-\left(\frac{r}{1-r}\right)^n\right) \qquad (3)$$

where $r(k/N)$ is the reward function. A detailed derivation can be found in Tung and Kleinrock (1995). Using this estimate, we obtain $\Phi(k)$:

$$\Phi(k)=\frac{\binom{N}{k}\frac{1}{1-2r(k/N)}\left(1-\left(\frac{r(k/N)}{1-r(k/N)}\right)^n\right)}{\sum_{k'=1}^{N}\binom{N}{k'}\frac{1}{1-2r(k'/N)}\left(1-\left(\frac{r(k'/N)}{1-r(k'/N)}\right)^n\right)} \quad (4)$$

For the reward function given in Fig. 2, $\Phi(k)$ obtained from a numerical test and theoretical estimation using Eq. (4) are compared in Fig. 3 for three different memory sizes, $n=5$, 10, 20 with $N=10$. As n increases, both numerical and theoretical estimates for the peak probability approach 1 at $k/N=0.3$. It can be easily seen from the theoretical estimation that as n approaches infinity, $\Phi(k)$ approaches a delta function located at k^*/N.

3. Extension for Multidimensional Application

The original Goore Scheme explained in the previous section is applicable only to one-dimensional problems. Consider an extension of the scheme to a multi-dimensional problem. Suppose there are L groups, each consisting of N_l players with $l=1, \cdots, L$. A reward function r is now a function of L parameters, which are the *yes*-vote percentages of each group. If there exists an optimum point in the L-dimensional space, $(k_1^*/N_1, k_2^*/N_2, \cdots, k_L^*/N_L)$, can we expect the same performance with the same automata (*i*, *e*., does the probability that the system stays at the optimum point approaches 1 after enough trials) The following two-dimensional reward function

$$r(k_1/N_1, k_2/N_2)=0.35\exp\left(-\left(\frac{k_1/N_1-k_1^*/N_1}{0.2}\right)^2-\left(\frac{k_2/N_2-k_2^*/N_2}{0.2}\right)^2\right)+0.3 \quad (5)$$

which has a peak value at $k_1^*/N_1=0.3$, $k_2^*/N_2=0.8$ as shown in Fig. 4 is chosen. For simplicity, we choose $N_1=N_2=N$. Now, a state is defined by a point in the two-dimensional parameter space spanned by $(k_1/N, k_2/N)$. Since the incident visiting a state in each dimension is independent of each other, the probability that $(k_1/N, k_2/N)$ state is visited is the product of probabilities that each state is visited in each dimension.

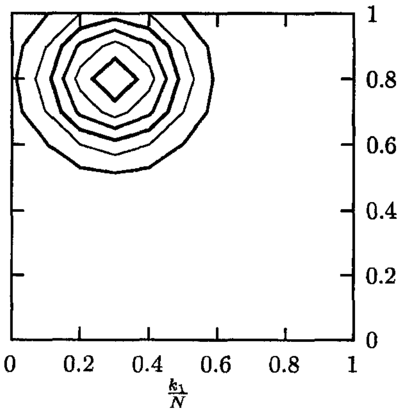

Fig. 4 Contour plot of the two-dimensional reward function, Eq. (5)

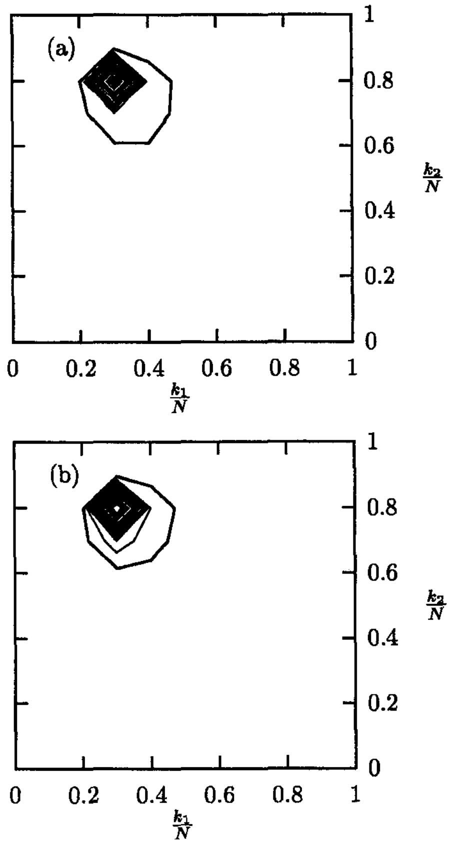

Fig. 5 Contours of the joint PDF between the *yes* percentages of each group in a two-dimensional problem. (a) Numerical result. (b) Theoretical estimation. The same contour levels are used with the level gap of 0.05

$$\Pi(k_1, k_2) = 2^{-2N}\binom{N}{k_1}\binom{N}{k_2} \tag{6}$$

If we can neglect the possibility of a state changing from $(k_1/N, k_2/N)$ to $((k_1 \pm 1)/N, (k_2 \pm 1)/N)$ simultaneously, whose probability approaches zero with N going to infinity, the time spent at the state can be estimated by the same method as in the one-dimensional problem. Therefore,

$$\tau_n(k_1, k_2) = \tau_n(r(k_1/N, k_2/N)) = \frac{1}{1-2r}\left(1-\left(\frac{r}{1-r}\right)^n\right) \tag{7}$$

This estimation of the persistence time for multi-dimensional problems is possible since the players are isolated from each other and do not know of each other's existence. Thus, the automaton of each player is influenced through r only. Using this, we can estimate the expectation probability $\Phi(k_1/N, k_1/N)$ as

$$\Phi(k_1/N, k_2/N) = \frac{\binom{N}{k_1}\binom{N}{k_2}\frac{1}{1-2r(k_1/N, k_2/N)}\left(1-\left(\frac{r(k_1/N, k_2/N)}{1-r(k_1/N, k_2/N)}\right)^n\right)}{\sum_{k_1'=0}^{N}\sum_{k_2'=0}^{N}\binom{N}{k_1'}\binom{N}{k_2'}\frac{1}{1-2r(k_1'/N, k_2'/N)}\left(1-\left(\frac{r(k_1'/N, k_2'/N)}{1-r(k_1'/N, k_2'/N)}\right)^n\right)} \tag{8}$$

As n increases, the probability density function approaches a delta function located at $(k_1^*/N, k_2^*/N)$. For the reward function of Fig. 4, a numerically obtained result is shown with the theoretically estimated distribution in Fig. 5. Here, $N=10$ and $n=10$. We intentionally choose $n=10$ for which the expectation probability distribution has a peak value of less than one. Also the case of $n=10$ has a finite area of nonzero probability value since an almost delta-function distribution is observed when $n=20$. The peak location is correctly captured in both theoretical and numerical results, and then distributions are similar. Without a loss of generality, it is straightforward to extend the analysis to problems with L greater than two. For an L-dimensional problem, the probability that $(k_1/N_1, \cdots, k_L/N_L)$ state is visited becomes

$$\Pi(k_1, \cdots, k_2) = 2^{-\Sigma_1^L N_i}\binom{N_1}{k_1}\cdots\binom{N_L}{k_L} \tag{9}$$

and the expectation probability becomes

$$\Phi(k_1/N_1, \cdots, k_L/N_L) = \frac{\binom{N_1}{k_1}\cdots\binom{N_L}{k_L}\frac{1}{1-2r(k_1/N_1, \cdots, k_L/N_L)}\left(1-\left(\frac{r(k_1/N_1, \cdots, k_1/N_L)}{1-r(k_1/N_1, \cdots, k_L/N_L)}\right)^n\right)}{\sum_{k_1=0}^{N_1}\cdots\sum_{k=0}^{N}\binom{N_1}{k_1'}\cdots\binom{N_L}{k_L'}\frac{1}{1-2r(k_1'/N_1, \cdots, k_L'/N_L)}\left(1-\left(\frac{r(k_1'/N_1, \cdots, k_L'/N_L)}{(1-r(k_1'/N_1, \cdots, k_L'/N_L)}\right)^n\right)} \tag{10}$$

4. Convergence Acceleration

In the previous section, it has been shown that as the memory size n increases, the peak probability increases. This just means that the time spent at the optimum state increases relative to the time spent at nonoptimum states. Furthermore, to achieve this, a much longer transient time should be spent before the system approaches an equilibrium state. For this scheme to be applied to a dynamical system which has a short dynamical time scale, the transient period should be minimized. Otherwise, the scheme will never catch up with the variation of the dynamical system at hand. To reduce the transient period without degradation of the peak equilibrium probability, we propose a variable memory size $n(t)$. Instead of keeping n fixed, we let n vary with the scheme's performance. For example, when the reward function increases, n increases as well, and vice versa. This allows the system to spend more time near the optimum state and to quickly divert from

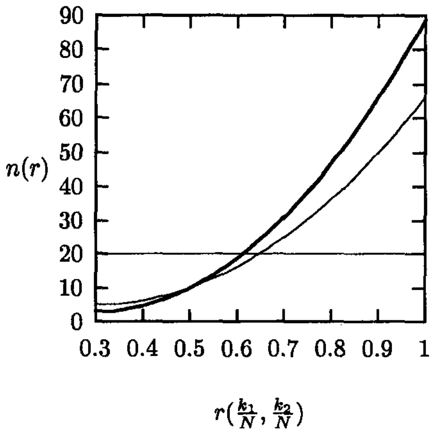

Fig. 6 Dynamic memory functions as functions of the reward function: dashed line, $n=20$; thin solid line, $n=5+5\left(\frac{r-0.3}{0.2}\right)^2$; thick solid line, $n=3+7\left(\frac{r-0.3}{0.2}\right)^2$

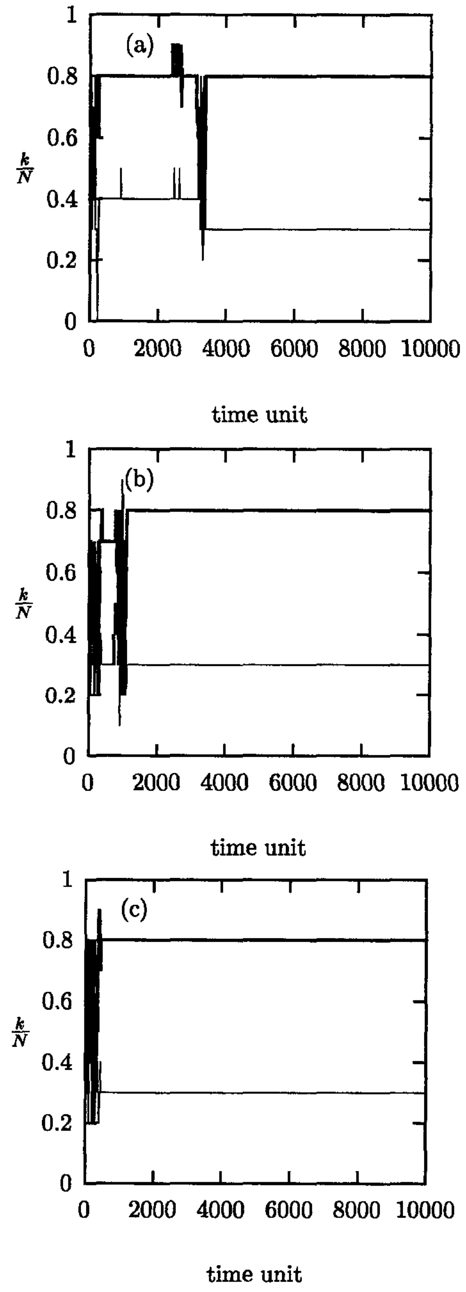

Fig. 7 Convergence performance of the yes percentage in a two-dimensional problem for 3 different dynamic memory functions: (a) $n=20$; (b) $n=5+5(r-0.3)/0.2)^2$; (c) $n=3+7((r-0.3)/0.2)^2$. Thin solid line denotes k_1/N and thick solid line denotes k_2/N

nonoptimum states. For the two-dimensional reward function of Fig. 4, we tested the following variable memory sizes including the case with fixed n (Fig. 6).

$$(i)\quad n=20 \tag{11}$$

$$(ii)\quad n=5+5\left(\frac{r-0.3}{0.2}\right)^2 \tag{12}$$

$$(iii)\quad n=3+7\left(\frac{r-0.3}{0.2}\right)^2 \tag{13}$$

Initial convergence behavior for the three cases of the same two-dimensional problem of the previous section is shown in Fig. 7 Compared to the constant n case, the systems with variable memories converge much more quickly to the optimum state ($k_1^*/N=0.3$, $k_2^*/N=0.8$) by an order of magnitude. Furthermore, the peak probability increases too. However, in cases with several local optima, the scheme might spend too much time in one optimum before finding a global optimum.

Next, we consider a one-dimensional problem with a large number of players N. With the memory size fixed at 20, as N increases from 10 to 40, the peak probability decreases as shown in Fig. 8(a). When $N=30$ and 40, even the peak location is not correctly predicted. This is due to the fact that as N increases, the resolution $1/N$ decreases so that it becomes more difficult to distinguish k^*/N from $(k^*\pm1)/N$. Increasing n cannot fix this problem since as long as n is larger than 20, the performance does not improve any further and even deteriorates with a longer transient period. Dynamic memory size was tested to mitigate this problem, resulting in a partial improvement as shown in Fig. 8(b). The correct peak location is captured, but the peak probability does not increase by much. To resolve this problem, the groups of 20, 30 and 40 players split into 2, 3, and 4 groups consisting of 10 players, respectively, and a multidimensional version of the scheme was applied with the result shown in Fig. 8(c). The reward function for each case is

$$r(k_1/N, k_2/N, \cdots, k_M/N)$$
$$=0.35\exp\left(-\sum_{m=1}^{M}\left(\frac{k_m/N-k_m^*/N}{0.2}\right)^2\right)+0.4 \tag{14}$$

with $M=1$ through 4 and $k_1^*/N=0.3$, $k_2^*/N=0.8$, $k_3^*/N=0.6$, $k_4^*/N=0.2$. Optimum peak locations for each case are correctly captured with

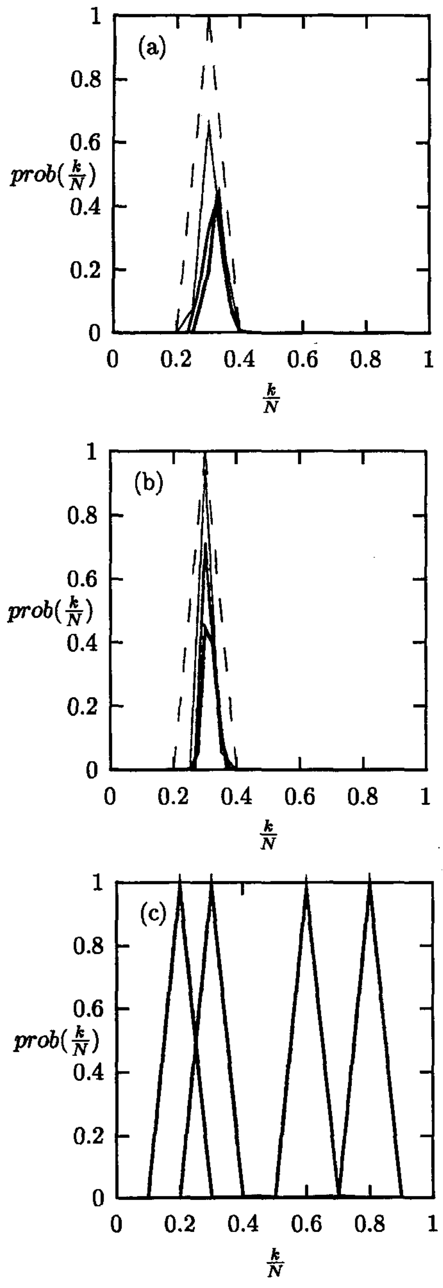

Fig. 8 PDF distribution of the *yes* percentage for (a) one-dimensional problem with different number of entities, 10(dashed), 20(thin solid), 30(solid) and 40(thick solid). The number of memory is kept constant at 20. (b) Effect of the dynamics memory function. The line styles are the same as in (a). (c) Effect of multidimensionalization up to 4. The number of entity is kept

a probability of almost 1. Therefore, the scheme performs much more efficiently when 40 players are split into 4 groups than when the 40 players remain in one group. By transforming a one-dimensional problem into a multi-dimensional problem the distance between states becomes shorter, and thus it takes less time to visit various points. Furthermore, an increase of $1/N$ from 1/40 to 1/10 helps the scheme to distinguish two adjacent states. This places a limitation on the resolution; N should not exceed 20. We also tested a case with moce than 4 groups. We found that the range of the optimum number of groups or parameters is less than 6.

5. Conclusion

We have investigated the performance of the original Goore Scheme and successfully extended the scheme to multi-dimensional problems. Splitting a one-dimensional problem with many possible states into a multi-dimensional problem with less states in each dimension improves the scheme's performance. To apply the scheme to a dynamical system with a short time scale, we improve the scheme by introducing a dynamical memory size. The improved scheme has been successfully applied to several sample problems. The number of players in a group should be less than 20, and the number of parameters should not exceed 5. This scheme can be applied to control of any dynamical system whose detailed mechanism is not known. Only the global output and control input are needed. The scheme finds the optimum input to produce the best output based on a very efficient trial-and-error technique.

Acknowledgements

We appreciate the financial support by KOSEF through Grant 981-1212-036-2. All computations were carried out at the KORDIC Supercomputing Center.

References

Bewley, T. and Liu, S., 1998, "Optimal and robust control and estimation of linear paths to transition," *J. Fluid Mech.,* Vol. 365, p. 305.

Bewley, T. and Moin, P., 1994, "Optimal control of turbulent channel flows," *Active Control of Vibration and Noise*, ASME DE-Vol. 75.

Choi, H., Moin, P. and Kim, J., 1994, "Active turbulence control for drag reduction in wall-bounded flows," *J. Fluid Mech.*, Vol. 262, p. 75.

Cortelezzi, L., Lee, K., Kim, J. and Speyer, J., 1998, "Skin-friction drag reduction via robust reduced-order linear feedback control," *Int. J. Comp. Fluid Dynamics*, Vol. 11.

Joshi, S. S., Speyer, J. L. and Kim, J., 1997, "A systems theory approach to the feedback stabilization of infinitesimal and finite-amplitude disturbances in plane Poiseuille flow," *J. Fluid Mech.*, Vol. 332.

Karlin, S. and Taylor, H. M., 1975, *A First Course in Stochastic Processes*, 2/e Academic Press, p. 45.

Koumoutsakos, P., 1999, "Vorticity flux control for a turbulent channel flow," *Physics of Fluids*, Vol. 11, No. 2, p. 248.

Lee, C. and J. Kim, 2001, "Application of the Goore Scheme to turbulence control for drag reduction (II) -Application to turbulence control-," *KSME Int.* J. Vol. 15 No. 11, pp. 1580-1587.

Lee, C., Kim, J., Babcock, D. and Goodman, R., 1997, "Application of neural networks to turbulence control for drag reduction," *Physics of Fluids,* Vol. 9, No. 6, pp. 1740-1747.

Lee, C., Kim, J. and Choi, H., 1998, "Suboptimal control of turbulent channel flow for drag reduction," *J. Fluid Mech.*, Vol. 358, p. 245.

Lee, K. H., Cortelezzi, L., Kim, J. and Speyer, J., 2001, "Application of reduced-order controller to turbulent flows for drag reduction," *Physics of Fluids*, Vol. 13, No. 5, pp. 1321-1330.

Tsetlin, M. L., 1964, *Finite Automata and Modeling the Simplest Forms of Behavior.* PhD thesis, V. A. Steklov Mathematical Institute.

Tung, B. and Kleinrock, L., 1993, "Distributed control methods," *Proceedings of the 2nd International Symposium on High Performance Distributed Computing*, Spokane, Washington, July 20-23.