공정시스템 연구회

최신 제어기법에의 기술 동향 고찰 : Simulation-Based Dynamic Programming

김 동 규*, 이 광 순**, 양 대 륙*
*고려대학교 화공생명공학과, **서강대학교 화공생명공학과

1. 서론

화학공학의 전통적 대상공정은 석유화학공정을 중심으로 한 연속공정이다. 이들 공정은 규모가 크고 공정물류가 연속으로 흐르기 때문에 수동으로 운전조건을 유지하는 것이 불가능하여 일찍부터 자동 제어가 도입되었다. Foxboro 사에서 1934년 최초로 상업용 공압식 PID 제어기를 생산한 것도 화학공장을 대상으로 한 것이었다. 그러나 운전방법 개선에 대한 요구와 1960년대 이후 제어기법의 비약적 발전에도 불구하고 화학공장의 보수적 특성으로 인하여, 화학공정은 전적으로 PID 제어기에 의존하는 단일루프 제어기법을 벗어나지 못하고 있었다. 그러던 것이 1979 Canada Shell 사에서 FCC (유동층 촉매 크래킹) 공정에 다변수 모델예측제어(MPC, Model-based Predictive Control)를 성공적으로 적용한 사례가 발표되며[1] 제어기법이 급격히 화학공정에 도입되기 시작하였다. 이 후 MPC는 사실상 화학공정 고급제어의 표준기법으로 자리를 잡았으며, 국내에도 이미 상당수의 MPC 프로젝트가 수행되었고 또 현재 수행 중에 있다. 이와 같이 MPC에 대한 높은 관심은 화학공정제어의 수준을 크게 끌어 올려 학술적 연구도 매우 활발하게 수행되어 1990년대 중반까지 MPC는 화학공정제어의 견인차 역할을 하였다[2, 3]. MPC에 대한 연구가 대체로 완성단계에 들어서며, 이후 화학공정제어에 관한 연구는 비선형제어, 회분공정제어 등 새로운 제어기법과 또한 전통적인 화학공정을 벗어나 반도체, 제지, 생물, 의료 등을 대상으로 다양화 되기 시작하였다.

한편 화학공학의 대상공정도 석유화학공정을 중심으로 한 전통적인 영역을 벗어나 생명기술(BT), 초미립 신소재 기술(NT) 등 새로운 영역이 급부상하기 시작하였으며, 전통적인 공정의 문제도 단위공정을 벗어나 전공장 최적화 및 운영 등 대규모 문제로 그 관심이 옮겨 가기 시작하였다. BT나 NT의 대상계도 규모는 작으나 분자 수준의 모델이 결합되어야 하는 multi-scale 모델링이 필요하기 때문에, 결국 화학공학의 대상 공정은 그 복잡성과 비선형성이 크게 증가되는 방향으로 이동하기 시작한 것이다. 이러한 공정의 최적제어는 결국 대규모 비선형 문제가 될 수 밖에 없으며, 기존의 기법들로는 그 해를 얻기가 어렵다. 이러한 화학공정제어의 패러다임의 변화에 대응할 수 있는 방법으로 simulation-based dynamic programming (SDP)이 새로운 주목을 받고 있으며 본 논문에서 그 개념을 소개하고자 한다.

제어, scheduling, planning과 같이 sequential 최적해를 요구하는 엔지니어링 문제들은 소위 functional optimization (FO)으로 최적화 문제가 표현되며, 모두 dynamic programming (DP) 문제로 변환이 가능하다. DP는 FO의 필요조건을 제공하는 minimum principle과 달리 충분조건을 제시한다는 장점이 있다. 특히 DP를 최적제어에 적용하는 경우에는 제어입력이 상태 되먹임 형태로 주어진다는 추가적인 이점이 있다. DP는 Bellman [4]이 1957년 처음 그 개념을 소개한 이후 White [5], Dreyfus와 Law [6], Whittle [7, 8], Ross [9] 등 최근 Bertsekas and Tsitsiklis [10]에 이르기 까지 많은 사람들의 연구과제가 되어 왔다. 그러나 DP는 변수의 차수가 조금만 늘어도 계산량이 급격히 증가하는 소위, 'curse of dimensionality' 문제를 가지고 있어 그 응용이 극히 제한되어 왔다. 이러한 이유로 제어에서는 준최적해를 제공하지만 실시간 응용이 가능한 MPC (Model-based Predictive Control)가 최적제어를 대신해 왔다[3]. 그러나 scheduling 등 비선형이 매우 심하고 정수변수가 포함된 문제에서는 마땅한 대안이 없었던 것이 사실이다. 최근 이러한 비선형 제어 문제들을 해결하고, DP의 'curse of dimensionality'의 문제점을극복할 수 있는 소위 neuro-dynamic programming (NDP)이라는 기법이 화공 분야에서 활발히 연구되기 시작하고있으며, DP의 중요성이 재부상하고 있다. NDP는 reinforcement learning (RL)이라는 이름으로 불려지기도 하며, 근사기를 사용하여 현재 상태에서 최적제어를 결정하는데 도움을 주는 방법들을 통칭하는 용어로 사용되고 있다. NDP는 'reinforcement learning (RL) on Samuel's checkers player' [11]나 'DP in the context of reinforcement learning' [12, 13], 'Heuristic dynamic programming' [14], 그리고 'explicit connection of RL to DP' [15] 등의 주제를 갖고 그 응용 결과들이 꾸준히 발표되었고 많은분야에 적용되어왔다. 최근에는 Kaisare et al. [16] 등이 NDP 알고리즘의 계산효율을 높이기 위하여 simulation을 이용한 SAE (Simulation-Approximation-Evolution) 알고리즘을 제안하였고, 이를 생물반응기에 적용하여 발표하기도 했다.

2. Neuro-Dynamic Programming (NDP)

2.1. Dynamic Programming (DP)

DP는 어떤 동특성 공정의 동적 최적화 문제의 해가 만족하여야 하는 충분조건을 제시해 준다. 아래의 이산시간 동특성 공정에 대해

$$x(k+1)=F(x(k),u(k)),\quad x(0)=x_0 \tag{1}$$

다음과 같은 성능지표를 최소화하는 제어벡터, $u(k)$, $k=0,\cdots,N-1$ 를 구하는 동적 최적화 문제를 생각하자.

$$J=\sum_{i=0}^{N-1}\phi(x(i),u(i))+\phi_N \tag{2}$$

여기서 ϕ는 현재 상태에서의 cost이고, ϕ_N은 최종 cost를 나타낸다. 적절한 입력을 결정하기 위해서는 적절한 성능지표의 정의가 필요하며, 최적제어의 경우는 흔히 다음과 같은 2차cost를 고려하는 경우가 많다.

$$\begin{aligned}\phi(x(k),u(k))&=\left(x(k+1)-x_{sp}\right)^T Q\left(x(k+1)-x_{sp}\right)+\Delta u(k)^T R\Delta u(k),\\ \phi_N&=\left(x(N+1)-x_{sp}\right)^T M\left(x(N+1)-x_{sp}\right)\end{aligned} \tag{3}$$

여기서 $\Delta u(k)=u(k)-u(k-1)$ 이고 x_{sp}는 설정점을 나타낸다.

식 (2)의 성능지표의 최적값은 상태의 초기조건 x_0에 따라 그 값이 달라질 것은 자명하다. 이제 이 개념을 확장하여 k 시점에서의 최적cost-to-go를 다음과 같이 정의하자.

$$J_k^*(x(k))=\min_u\left[\sum_{i=k}^{N-1}\phi(x(i),u(i))+\phi_N\right] \tag{4}$$

J_k^*는 당연히 상태변수 $x(k)$ 의 함수가 된다. 이제 식 (4)는 다음과 같이 최적 cost-to-go에 관한 식으로 재정리된다.

$$\begin{aligned}J_k^*(x(k))&=\min_u\left[\phi(x(k),u(k))+\sum_{i=k+1}^{N-1}\phi(x(i),u(i))+\phi_N\right]\\&=\min_u\left[\phi(x(k),u(k))+J_{k+1}^*(x(k+1))\right]\\&=\min_u\left[\phi(x(k),u(k))+J_{k+1}^*(F(x(k),u(k)))\right],\\ J_N^*(x(N))&=\phi_N\end{aligned} \tag{5}$$

이 식을 Bellman 방정식이라 부르며, 이 방정식을 만족시키는 $u(k)$ 가 $x(k)$ 의 함수로 결정되므로 자연스럽게 되먹임 형태의 제어가 얻어진다. N이 무한대이고 F가 시불변인 경우, 최적 cost-to-go는 상태만의 함수가 될 것이며, 이 때 Bellman 방정식은

$$\begin{aligned}J^*(x(k))&=\min_u\left[\phi(x(k),u(k))+J^*(x(k+1))\right]\Rightarrow\\ J^*(x)&=\min_u\left[\phi(x,u)+J^*(F(x,u))\right]\end{aligned} \tag{6}$$

와 같이 간략화 된다.

Bellman 방정식의 해 $J^*(x)$를 해석적으로 얻는 것은 특별한 경우 (예, 선형 동특성과 2차 cost)를 제외하고는 해석해를 구할 수 없기 때문에, 일반적으로 수치해를 구하게 된다. 식 (5)의 경우는 $k=N$에서 출발하여backward로 풀어 내려 오는데, 이때 매 시점 마다 $J^*(x(k))$을 모든 가능한 상태 값에 대해 구해 놓아야 하므로 이 수치해는 상태변수 x의 차수가 증가하면 계산량이 지수적으로 급증하게 된다. 식 (6)에서도 $J^*(x)$를 알기 위해서는 모든 가능한 x값에 대한 상당량의 계산이 요구되며 상태변수 x의 차수가 증가하면 계산량이 지수적으로 급증하게 되는 동일한 문제를 가진다. 이 문제를 'curse of dimensionality' 라고 부르며 이로 인해 DP는 사실상 적용하기 어려운 방법으로 인식 되어왔다.

2.2. Simulation-based dynamic programming

이러한 'curse of dimensionality'의 문제점을 극복하기 위해 NDP가 개발되었으며 화학공학 분야에서도 그 관심이 점증되고 있다. NDP는 RL (reinforcement learning)이라 부르기도 하며, $J^*(x)$를 simulation 혹은 실험 데이터에 의존하여 학습하는 방법을 통칭한다.

NDP의 적용을 효율적으로 수행하는 방법의 하나로, Kaisare 등이 simulation에 근거한 소위 Simulation-Approximation-Evolution (SAE) 알고리즘을 제안하였다[16]. 이 방법은 최적 cost-to-go $J^*(x)$를 준최적 상태의 simulation 데이터를 이용하여 학습시키며, 계속 개선시켜 나가는 방법으로, 그림 1에 SAE의 개념을 보였다[17].

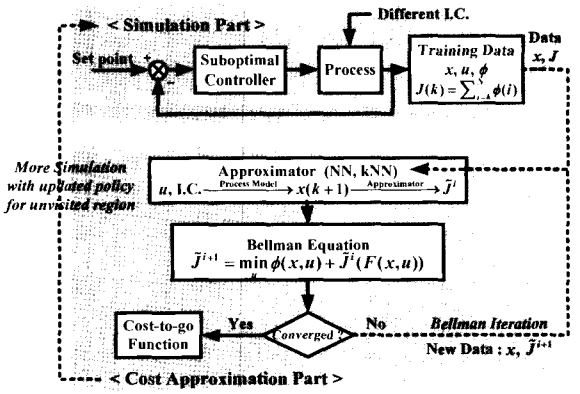


그림 1. Architecture for offline computation of cost-to-go approximation.

SAE 알고리즘은 크게 "Simulation Part"와 "Cost Approximation Part"의 두 부분으로 나뉘어 진다. 각 상태에서 최적제어를 결정하기 위해선 식(6)의 최적 cost-to-go $J^*(x)$를 얻어야 되고, 이를 위해서는 여러 상태 x에 대한 최적 cost-to-go 값들과 이 들을 함수의 형태로 계산할 수 있는 근사기(arppoximator)를 구하는 것이 필수적인 요소이다. Simulation part는 바로 이 근사기를 구하기 위한 다양한 조건의 데이터를 생성시키는 부분이다. 그러나 비선형 공정에 대한 최적해는 일반적으로 구할 수 없으므로 선형화 공정에 대한 최적제어, 혹은 모델예측제어 등을 여러 초기조건에 대해 적용하여 준최적 입력, $u(t)$ 및 상태값, $x(t)$ 들을 얻는다. 이 때 visit된 상태값 들을 초기값으로 하여 근사 최적 cost-to-go를 식(7)과 같이 계산한다[16, 17].

$$J(x) = J(x(k)) = \sum_{i=k}^{N} \phi(i) \tag{7}$$

여기서 N은 새로운 정상상태에 도달할 정도로 커야 된다. SAE알고리즘의 두 번째 부분은 "Cost Approximation Part"이다. 이 부분은 simulation part에서 구한 데이터들로부터 cost-to-go 함수와 상태변수간의 관계를 근사하는 부분이다. 이를 위해서는 먼저 x와 J사이의 관계를 표현할 수 있는 신경회로망과 같은 근사기가 필요하다. 이렇게 근사된 관계를 $\tilde{J}(x)$라 하자. 이제 최적 cost-to-go가 식 (6)의 Bellman방정식을 만족하여야 한다는 사실을 이용하여, $\tilde{J}(x)$가 Bellman방정식을 만족하도록 다음과 같은 Bellman iteration을 $\tilde{J}^l(x)$가 수렴할 때까지 수행한다[16, 17].

$$\tilde{J}^{l+1}(x) = \min_{u} \left[\phi(x,u) + \tilde{J}^l\left(F(x,u)\right)\right] \tag{8}$$

SAE 알고리즘에서 사용되는 근사기의 성능은 제어기의 성능에 결정적인 역할을 한다. 근사기는 크게 총괄 근사기(global approximator)와 국부 근사기(local approximator)의 두 종류가 사용된다. 총괄 근사기는 비선형 회귀모델을 말한다. 총괄 근사기로 가장 많이 사용되는 신경회로망은 외삽이 가능하나 그 값의 신뢰도가 낮고 학습에 많은 데이터와 시간을 필요로 한다는 단점이 있다. 또한 Bellman iteration의 수렴을 보장할 수 없다는 문제점도 갖고 있다. 그러나 일단 학습이 되면 계산 속도가 매우 빠르고, 학습데이터가 별도로 보관될 필요가 없다는 장점이 있다. 반면에 국부 근사기 중 가장 대표적인 k-nearest neighbor(kNN)는 근사하고자 하는 점과 가장 가까운 위치에 존재하는 k개의 지점을 학습용 데이터에서 찾아 이를 이용하여 근사하는 방법이다. kNN은 복잡한 비선형성도 쉽게 다룰 수 있고, 학습에 들어가는 시간과 노력이 거의 없다는 장점이 있다. 신경회로망에 비하여 Bellman iteration의 수렴 기능성이 높은 것으로 보고되어 있으나, 학습용 데이터의 양이 많아지면 해당 데이터를 찾는데(querying) 요구되는 계산량이 많아진다. 또한 querying을 위해서 방대한 양의 데이터를 항상 보관하고 있어야 되는 단점이 있다[18, 19].

SAE와 같이 simulation에 근거한 NDP방법을 simulation-based dynamic programming (SDP)라 하는데, 이제 SDP의 몇 가지 적용 예를 살펴 보고자 한다.

3. Published Examples

3.1. Playing Tic-Tac-Toe

2000년, R.S. Sutton의 Reinforcement Learning 책에[20] 의하면, 우선 간단한 NDP적용 예로써 Tic-Tac-Toe 게임을 들고 있다(그림 2).

X	O	
	X	

그림 2. Playing Tic-Tac-Toe

이 경기는 그림 2와 같이 주어진 3×3의 상태 공간(state space)안에 X 혹은 O를 채워가는 경기로, X나 O 3개를 이용하여 가로나, 세로 혹은 대각선을 먼저 채우면 승리하는 경기이다. 이 경기에 NDP를 적용하기 위해서는, 우선 경기에 대한 가능한 시스템 상태의 리스트를 작성해야 한다. 각 상태(state)에서 이길 가능성이 가장 높은 초기 예측값에서 경기를 시작하고, 현재 상태(x_k)에서 이길 가능성이 가장 높은 상태로 옮겨가도록 행동을 선택한다. 상대방이 수를 둔 다음에 옮겨간 상태를 (x_{k+1})이라고 할 때, 이 때 이길 가능성의 값을 근거로 하여 현재상태의 값을 식(9)와 같이 수정하게 된다[20].

$$v^{new}(x_k) = v^{old}(x_k) + \alpha\left(v^{old}(x_{k+1}) - v^{old}(x_k)\right) \tag{9}$$

경기자가 항상 이기는 것을 선택하리라고는 기대하지 못하지만, 각기 다른 상태에서 다른 행동을 선택하도록 배워서 다양한 행동패턴을 익히는 것이 중요하다. 다양한 행동의 패턴을 데이터로 갖게 되면, 하나의 상태에 위치했을 때 다음에 어떤 행동을 취해야 가장 이길 가능성이 높은지를 판단하여 그 방향으로 나아갈 수 있다. 이처럼 현재 상태로부터 입력값에 따라서 나아갈 방향을 결정하여 그 목적에 맞는(이 경우에는 이길 가능성이 가장 높은 것을 찾는 것이 목적이다.) 입력의 패턴을 찾아 원하는 목적을 이루는 것이 NDP의 가장 기본이 되는 것이다. 조금 복잡한 경기로 확장한 경우로서 chess 경기를 들 수 있는데, 이 경기는 경우

의 수가 훨씬 많고 복잡하지만, 역시 NDP 적용의 한 예로서 다수의 논문이 발표된 바 있다[10, 11, 19].

3.2. pH Neutralization Process

경우의 수로서 진행되지 않는 예로서 일반적인공정의 제어를 들 수 있는데, 본 논문에서는 2003년 Kim *et. al.*이 논문에서 발표한 pH 중화공정을 예로 들기로 한다[17, 22]. pH 중화공정은 중화점 근처에서 비선형성이 심하고, 시간에 따라 그 성격이 변하므로, 오랫동안 비선형 화학공정에 대한 벤치마크로 널리 사용되어 왔다. 대상이 되는 pH 중화공정은 그림 3과 같고, 공급물(q_1)로 HNO_3의 강산을 사용하였고, 완충용액(q_2)으로는 $NaHCO_3$의 약산을 사용하였다. 또한 적정용액(q_3)으로 NaOH와 $NaHCO_3$가 섞인 강염기를 사용하였다. 이 공정은 25°C상에서 이온이 완전 해리되고, 용액이 완전혼합이 된다는 가정을 갖고 조업되며, 펌프나 센서의 동특성은 무시된다고 가정하였다. 이 때의 조업 조건은 표 1에 나타나있다[21].

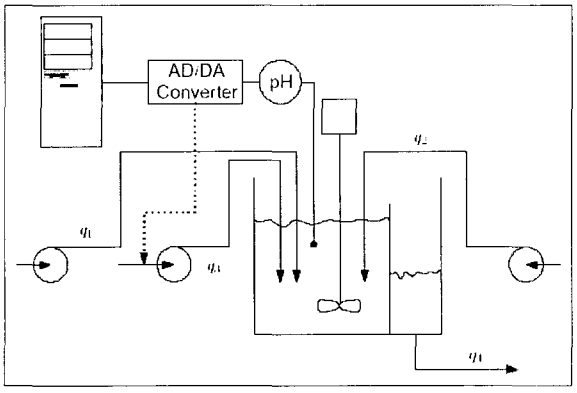


그림 3. The pH neutralization process.

표 1. Operating conditions of pH neutralization process.

Symbols	Values	Symbols	Values
V	2500 [ml]	$[q_1]$	0.003 M HNO_3 5.0×10^{-5} M H_2CO_3
q_1	9.0 [ml/s]		
q_2	0.6 [ml/s]	$[q_2]$	0.01 M $NaHCO_3$
q_3	8.5 [ml/s]	$[q_3]$	0.003 M NaOH 5.0×10^{-5} M $NaHCO_3$

일반적으로 강산 강염기 반응은 수용액에서 순간적으로 평형에 도달하고, 이것은 반응속도가 무한대의 값을 갖는 것을 의미한다. 이 때 반응속도 항은 공정모델에서 무시하여 공정모델을 단순화 할 수 있고, 이러한 접근에서 반응불변량(reaction invariant) 개념을 도입하여 상태방정식을 구성하면 다음의 식(10)과 같이 나타내어진다. 반응불변량 W_a는 전하에 관련된 값이고, W_b는 탄소이온에 관련된 값이다[21].

$$\begin{aligned} \dot{x} &= f(x,t)+g(x,t)u+F_\theta(t)\theta \\ c(x,y) &= 0 \end{aligned} \tag{10}$$

여기서 각 변수는 다음과 같이 정의된다.

$$f(x,t)=\frac{1}{V}\begin{bmatrix} q_2(W_{a2}-x_1)-q_1x_1 \\ q_1(W_{b1}-x_2)-q_2x_2 \end{bmatrix},\quad g(x,t)=\frac{1}{V}\begin{bmatrix} W_{a3}-x_1 \\ W_{b3}-x_2 \end{bmatrix},$$

$$F_\theta(t)=\frac{1}{V}\begin{bmatrix} q_1 & 0 \\ 0 & q_2 \end{bmatrix},\quad \theta=\begin{bmatrix} W_{a1} & W_{b2} \end{bmatrix}^T,\quad x=\begin{bmatrix} W_{a4} & W_{b4} \end{bmatrix}^T,$$

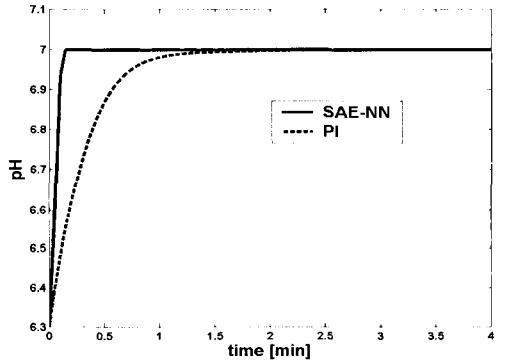
$$u=q_3,\quad y=\mathrm{pH}_4,\quad pK_1=-\log K_{a1},\quad pK_2=-\log K_{a2},$$

$$W_{a1}=[\mathrm{H}^+]_t-[\mathrm{OH}^-]_t-[\mathrm{HCO}_3^-]_t-2[\mathrm{CO}_3^{2-}]_t,$$

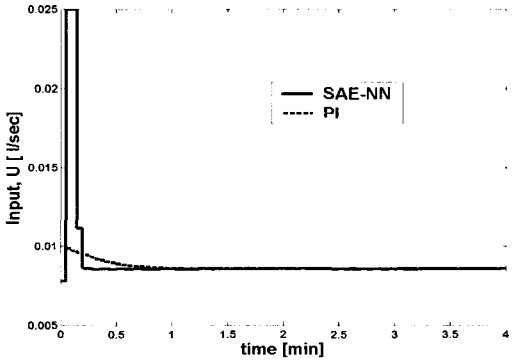
$$W_{b1}=[\mathrm{H_2CO_3}]_t+[\mathrm{HCO}_3^-]_t+[\mathrm{CO}_3^{2-}]_t.$$

$$c(x,y)=x_1+10^{y-14}-10^{-y}+x_2\frac{1+2\times10^{y-pK_2}}{1+10^{pK_1-y}+10^{y-pK_2}}=0.$$

위의 pH 중화공정에 대하여, SAE 알고리즘을 적용하여 제어를 한 경우 다음과 같은 결과를 얻을 수 있다. 총괄 근사기인 신경회로망을 이용한 경우 그림 4에서 보듯이 빠르고 원만하게 설정점으로 접근하는 것을 볼 수 있는데, 잘 조율된 PI 제어기보다 훨씬 좋은 제어결과를 보임을 볼 수 있다[17, 22].



(a) pH change



(b) Input chang

그림 4. Comparison of results between PI control and SAE by NN with respect to a step change in set point (pH 6.37).

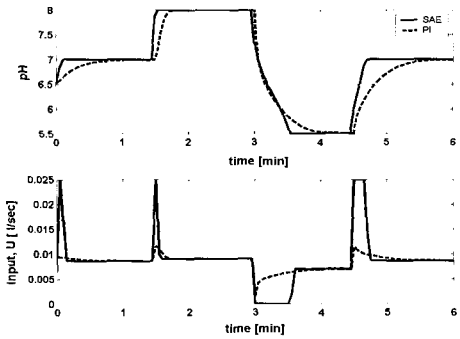
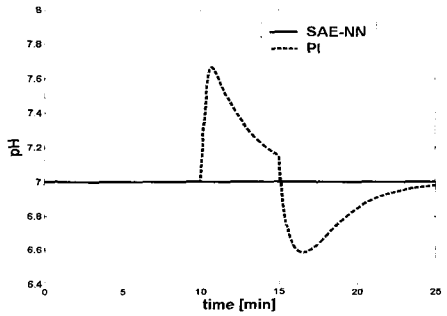
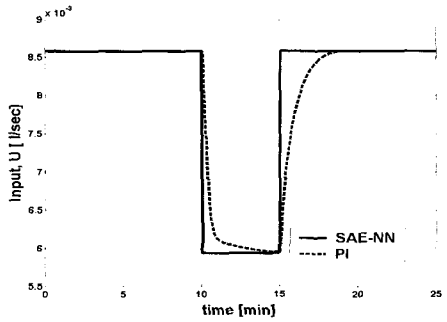


그림 5. Comparison of results between PI control and SAE by NN with respect to multi-step set point change.

그림 5에서 보듯이 설정점을 여러 차례 바꿔가면서 설정값측정 제어(servo control)을 한 경우에도 PI제어기 보다 좋은 성능을 보임을 볼 수 있고, 외란에 대한 정치 제어(regulatory control)의 경우에도 (그림 6) 잘 조율된 PI 제어기 보다 신경회로망을 이용한 SAE 알고리즘이 훨씬 좋은 제어 결과를 보이는 것을 볼 수 있다[17, 22].



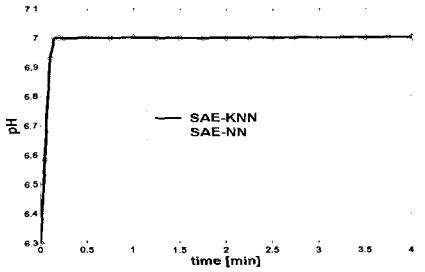
(a) pH change



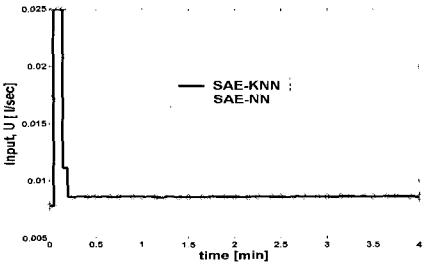
(b) Input chang

그림 6. Comparison of results between PI control and SAE by NN with respect to disturbance.

그림 7의 경우에는 neural network의 총괄 근사기 대신에 국부 근사기인 k-nearest neighbor방법을 사용한 경우인데, k-nearest neighbor방법은 학습에 대한 부담을 줄이면서 신경회로망을 썼을 때와 같은 효율을 보여주고 있음을 볼 수 있다[17, 22].



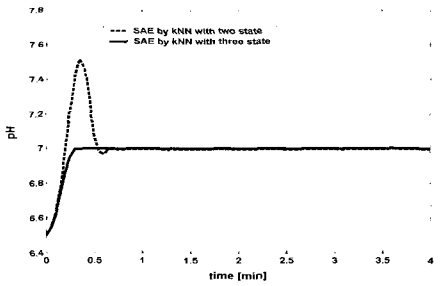
(a) pH change



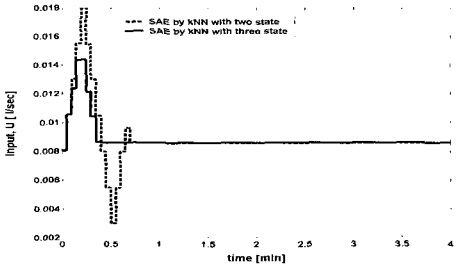
(b) Input chang

그림 7. Comparison of results between SAE by kNN and SAE by NN with respect to set point change (pH 6.37).

입력 값 u에 제약조건이 있는 경우에는 기존의 상태변수(state variable)만으로는 cost-to-go 값을 충분히 알 수 없으므로, u에 대한 정보를 상태변수로서 추가하여 NDP 알고리즘을 적용하는 것이 올바른 방법으로 u를 상태변수로 추가하지 않은 경우보다 나은 결과를 보였고(그림 8), 이 경우 외란에 대한 정치 제어도 잘 되고 있음을 볼 수 있다(그림 9) [22].

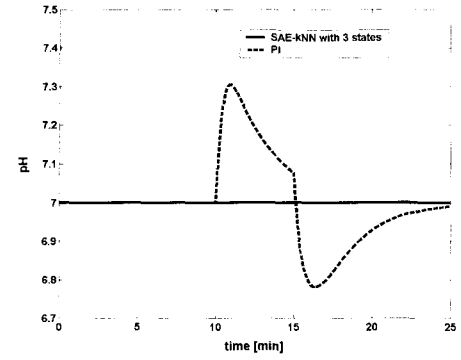


(a) pH change

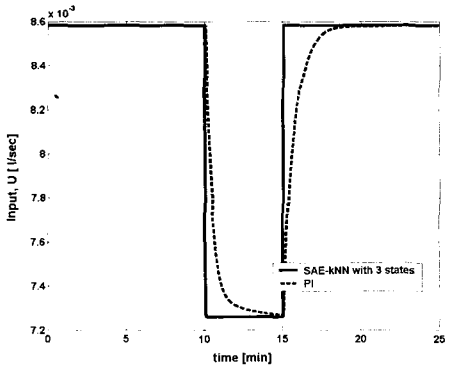


(b) Input change

그림 8. Comparison of results between with using two states and with using three states with a restriction on u (Δu_{max} = 0.0025 [l/sec], Unrestricted case Δu_{max} = 0.025 [l/sec])



(a) pH change



(b) Input change

그림 9. Comparison of results between PI control and SAE by kNN with respect to disturbance (A decrease of 15% in $W_{n1}^{'}$)

3.3. Microbial Cell Reactor

2003년 Kaisare et. al.가 발표한 논문에서는 다중정상상태를 가진 생물반응기의 생성물 농도 제어의 결과를 발표하였다[16]. 이 결과를 살펴보면, Klebsiella oxytoca를 생산하기 위한 생물 반응기는 glucose와 arabinose를 영양소로 넣어주며, 낮은 정상상태와 높은 정상상태의 두 가지 정상상태를 갖는 공정이다(그림 10). 이 공정에서는 희석농도(D)를 조절함으로써 생산 균체의 농도(C)를 높은 정상상태로 유지되게 하는 것을 제어의 목적으로 하고 있다[16].

표 2. Key variables and parameters of the system.

State Variables	s_1	Glucose (gm/L)
	s_2	Arabinose (gm/L)
	e_1	Key enzyme(s)-1 (gm/gm dry wt.)
	e_2	Key enzyme(s)-2 (gm/gm dry wt.)
	c	Biomass (gm/L)
Manipulated Variable	D	Dilution rate (hr^{-1})
Controlled Variable	c	Biomass
Parameter	s_{2f}	$s_2$feed rate (gm/L)

표 2에는 시스템의 주요 변수 및 파라미터 들이 나와있고, 이 때 사용되어진 상태(state)는 추가 상태변수인 s_2의 공급속도를 포함하여 6개의 상태변수($x=[s_1 s_2 e_1 e_2 c\ s_{2f}]'$)를 사용하였다.

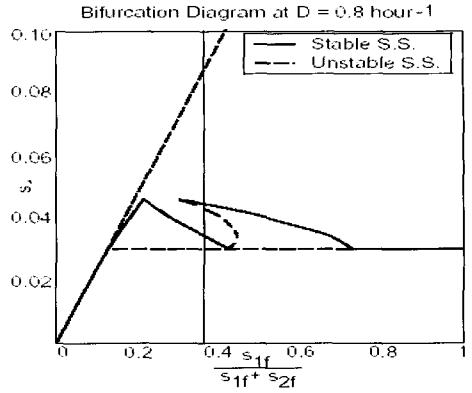


그림 10. Steady state bifurcation diagram for Klebsiella Oxytoca growing on glucose and arabinose

시스템은 다음과 같은 식(11)이 각 구성성분에 대하여 세워져서 5개의 ODE 모델식으로 구성된다. 이 때 시스템의 이단계 적응 생장(diauxic growth)을 표현하기 위하여 cybernetic modeling을 이용한 모델이 사용되었다[23].

$$\begin{aligned}
\frac{dS_i}{dt} &= D\left(S_{if}-S_i\right)-\left(r_i v_i\right)Y_i c \qquad i=1,\ 2... \\
\frac{de_i}{dt} &= \left(r_{e_i} u_i\right)-\beta e_i - r_g e_i + r_{e_i}^{*} \qquad i=1,\ 2... \\
\frac{dc}{dt} &= \left(r_g - D\right)\ c
\end{aligned} \tag{11}$$

여기서, $u_1^{sc}=\frac{r_1}{r_1+r_2}$, $v_1^{sc}=\frac{r_1}{\max(r_1,r_2)}$, $r_i=\mu^{\max}\frac{S_i}{K_i+S_i}\left[\frac{e_i}{e_i^{\max}}\right]$, $r_{e_i}=\alpha_i\frac{S_i}{K_{e_i}+S_i}$, $r_g=r_1v_1+r_2v_2$.

SAE알고리즘에서 사용된 single stage cost는 균체의 농도를 정의하는 $x(5)$의 농도와 그 설정값 간의 차를 사용하여 식(12)와 같이 정의하였다.

$$\phi(x,u)=Q\left\{r-x(5)\right\}^2+R\left\{\Delta u\right\}^2 \tag{12}$$

이 시스템에 SAE알고리즘을 적용하여 제어를 행한 결과는 그림 11 및 표 3에서 slMPC (successive linearization based nonlinear Model Predictive Control)와 비교하였다[16]. 이 결과에서 SAE알고리즘을 사용한 것이 slMPC에 비하여 훨씬 나은 결과를보여줌을 볼 수 있다. 또한 cost-to-go 근사값이 원래의 cost-to-go의 근사값에 대한 학습자료의 범위를 벗어나게 되면, 이를 이유로 SAE알고리즘이 원하는 성능을 보이지 않는 경우가 발생하는데, 학습되지 않은 영역으로 공정이 벗어나지 않도록 목적함수에 제한을 가하거나, 추가적인 모사를 수행하여 얻은

학습자료를 cost-to-go 값의 근사에 반영함으로써 SAE 알고리즘의성능을 향상시킬 수 있다. 이러한 수정을 가한 SAE알고리즘이 계산 시간도 짧고, 비용(cost)도 작은 좋은 결과를 얻을수 있음을 보고하고 있다(그림 11, 표 3, [16]).

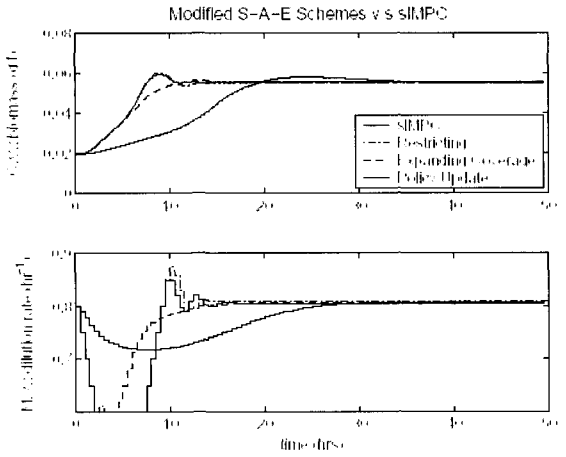


그림 11. Performance of the modified schemes as compared to slMPC control law (thick line)

표 3. Details of successively linearized MPC algorithm v/s S-A-E scheme and its modifications.

Control Algorithm	Number of data points	Cost Iterations	Number of hidden nodes	Total cost (at x(0))	CPU Time (seconds)
SIMPC	-N.A.-	-N.A.-	-N.A.-	22.54	1080.3
S-A-E Scheme	1200	4	5	24.18	98.7
w/ Restricting	1200	2	4	9.06	127.7
w/ Add Sim	2088	4	5	9.37	79.5
w/ Policy update	1395	7	9	10.32	74.12

4. NDP응용의 확장 및 그 전망

엔지니어링 문제에서 DP의 중요성은 새삼 강조할 필요가 없다. 그러나 감당할 수 없는 계산량으로 그 응용이 극히 제한적이었던 것이 사실이다. 이러한 DP의 한계를 허물 수 있는 수치적 해법으로 NDP가 개발되고 연구 되어왔으며, 현재로서 그 전망은 매우 긍정적이다. NDP는 이미 금융, 재고관리, SCM (Supply Chain Management), 통신 네트워크 등 다양한 분야에서 성공적으로 응용 되어왔으며 그 응용 분야는 계속 넓어지고 있다. 특히 Samuel's checkers player [11], backgammon 경기 [24, 25, 26], acrobat [27], elevator dispatching [28] 과 같은 대표적인 benchmark 문제에서도 기존의 해법으로는 얻을 수 없던 좋은 결과를 제시하고 있다.

서론에서 언급한 바와 같이, 화학공학은 이미 종래의 틀을 벗어나 그 복잡성이 크게 증대된 문제를 대상으로 하고 있다. 이와 같은 대상 공정의 최적제어도 기존의 기법으로는 해를 구하기 어려운 심한 비선형성 또는 대규모 문제로 나타나고 있다. NDP는 이러한 문제를 다룰 수 있는 긍정적인 해법이며 이미 생물반응기의 제어 [16] 뿐만이 아니라 다양한 비선형공정에 대한 제어 적용 [18, 22], 스케줄링을 통한 신약개발 의사 결정 [19] 등을 통하여 그 효용성을 증명한 바 있다. 아직 multi-scale 문제에 대한 NDP의 적용은 보고된 바 없으나 그 응용이 매우 기대된다. 그러나 화학공학에서의 NDP응용은 이제 막 발을 내 디딘 수준이다. NDP는 근사적 수치해를 주는 기법으로, 수렴성의 개선을 위해서는 앞으로도 많은 경험과 기법들이 개발되어야 한다[18, 19]. 많은 화학공학도들이 NDP 에 관심을 갖고 연구에 참여하여 그 적용범위가 더욱 확장되고 기법이 더욱 다듬어 지기를 기대해 본다.

참고문헌

1. Cutler, C. R., & Ramaker, B. L., "Dynamic matrix control-a computer control algorithm," *In Proceedings of the joint automatic control conference*, 1980.
2. Garcia, C.E., Prett, D. M., & Morari,M., "Model preditive control: Theory and practice-a survey," *Automatica*, 25(3), pp. 335-348, 1989.
3. S. Joe Qin, Thomas A.Badgwell, "A survey of industrial model predictive control technology," *Control Engineering Practice*, 11, pp. 733-764, 2003.
4. R. E. Bellman, "Dynamic Programming," Princeton University Press, New Jersey, 1957.
5. White, D. J., "Dynamic Programming," Holden-Day, San Francisco, 1969.
6. Dreyfus, S. E. and Law, A. M., "The Art and Theory of Dynamic Programming," Academic Press, New York, 1977.
7. Whittle, P., "Optimization over Time," vol. 1, Wiley, NY, 1982.
8. Whittle, P., "Optimization over Time," vol. 2, Wiley, NY, 1983.
9. Ross, S., "Introduction to Stochastic Dynamic Programming," Academic Press, New York, 1983.
10. Dimitri P. Bertsekas and John N. Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, Massachusetts, 1996.
11. M. Minsky, "Steps Toward Artificial Intelligence," *Proc. Of the IRE*, vol. 49, no. 1, pp. 8-30, January. 1961.
12. Andreae, J. H., "A learning machine with monologue," *International Journal of Man-Machine Studies*, 1, pp. 1-20., 1969.

13. Andreae, J. H., "Learning machines-a unified view," In Meetham, A. R. and Hudson, R. A., editors, *Encyclopedia of Information, Linguistics, and Control*, pp. 261-270. Pergamon, Oxford, 1969.
14. Werbos, P. J., "Advanced forecasting methods for global crisis warning and models of intelligence," *General Systems Yearbook*, 22, pp. 25-38, 1977.
15. Watkins, C. J. C. H., "Learning from Delayed Rewards," *Ph.D thesis*, Cambridge University, Cambridge, England, 1989.
16. Niket S. Kaisare, Jong Min Lee and Jay H. Lee, "Simulation based strategy for nonlinear optimal control : Application to a microbial cell reactor," *Int. J. Robust Nonlinear Control*, pp. 347-363, 2003.
17. Dong Kyu Kim, Dae Ryook Yang, "Control of pH Neutralization Process using Simulation Based Dynamic Programming," *ICCAS 2003 in Gyeongju*, 629, 2003.
18. Jong Min Lee, Niket S. Kaisare, Jay H. Lee, "Simulation Based Dynamic Programming Strategy for Improvement of Control Policies," *AIChE 2003 annual meeting in San Francisco*, 438c, 2003.
19. Jay H. Lee, "Simulation Based Dynamic Programming for Process Control, Optimization and Scheduling," *AIChE 2003 annual meeting in San Francisco*, 434f, 2003.
20. Sutton, R.S. and Barto, A.G., "Reinforcement Learning," MIT, 2000.
21. Ahrim Yoo, "Experimental Parameter Identification and Control of pH Neutralization Process Based on an Extended Kalman Filter," *Master Thesis*, Korea University, 2002.
22. Dong Kyu Kim, Kwang Soon Lee, Dae R. Yang, "Control of pH Neutralization Process Using Neuro Dynamic Programming," *AIChE 2003 annual meeting in San Francisco*, 433e, 2003.
23. D. S. Kompala, D. Ramkrishna, N. B. Jansen, and G. t. Tsao., "Investigation of bacterial growth on mixed substrates: Experimental evaluation of cybernetic models," *Biotechnology and Bioengineering*, 28, pp.1044-1055, 1986.
24. Tesauro, G. J., "Practical issues in temporal difference learning," *Machine Learning*, 8, pp. 257-277, 1992.
25. Tesauro, G. J., "TD-gammon, a self-teaching backgammon program, achieves master-level play," *Neural Computation*, 6(2), pp. 215-219, 1994.
26. Tesauro, G. J., "Temporal difference learning and TD-Gammon," *Communications of the ACM*, 38, pp. 58-68, 1995.
27. Spong, M. W., "Swing up control of the acrobat," *In Proceedings of the 1994 IEEE Conference on Robotics and Automation*, San Diego, CA, 1994.
28. Crites, R. H. and Barto, A. G., "Improving elevator performance using reinforcement learning," In D. S. Touretzky, M. C. Mozer, M. E. H., editor, *Advances in Neural Information Processing Systems: Proceedings of the 1995 Conference*, pp. 1017-1023, Cambridge, MA. MIT Press, 1996.

저 자 소 개

《이광순(李光淳)》

- 1955년 3월 23일생.
- 1973년 서울대학교 화학공학과 졸업 (학사).
- 1979년 KAIST 화학공학과 졸업(석사).
- 1983년 KAIST 화학공학과 졸업 (공학박사).
- 1983년 현재 서강대학교 화공생명공학과 교수.
- 1986년~1987년 카나다 워털루 대학교 방문교수.
- 1994년~1995년 미국 Auburn 대학교 방문교수.

《양대륙(梁大陸)》

- 1958년 9월 4일생.
- 1981년 서울대학교 화학공학과 졸업(학사).
- 1982년 KAIST 화학공학과 졸업(석사).
- 1983년~1986년 KIST Process Dev. Lab. 연구원.
- 1990년 미국, Univ. of California, Santa Barbara, 화학공학과 졸업(공학박사).
- 1990년~1991년 ABB Simcon Inc., U.S.A. Application Engineer.
- 1992년~1994년 포항공대 공과대학 화학공학과 조교수.
- 1994년 현재 고려대학교 공과대학 화공생명공학과 교수.
- 2000년~2001년 Georgia Institute of Technology, USA, 교환교수.

《김동규(金東奎)》

- 1972년 8월 25일생.
- 1996년 고려대학교 화학공학과 졸업 (학사).
- 1998년 동대학원 졸업(석사).
- 1998년 현재 동대학원 박사과정.