

## 학부 졸업논문 작성에 관한 조언

김도형(성신여자대학교 IT학부)

### 0. 들어가는 말

이 글은 성신여자대학교 컴퓨터정보학부를 주전공 또는 복수전공으로 하여 졸업논문을 준비하는 4학년 학생들을 위한 것입니다. (2009년도부터 IT학부로 개편된 체제에서의 졸업 기준은 다소 변동이 있을 수 있으므로, 이 글에서의 조언이 그대로 적용되지 않을 가능성이 있습니다. 그러나 만약 IT학부에서도 ‘졸업논문’을 요구한다면 여기에서의 조언 대부분을 참고할 수 있을 법합니다.)

이 글의 내용 대부분은 제가 수업 시간이나 논문 지도시에 이미 구두(口頭)로 학생들에게 전달한 바 있습니다만, 그럼에도 불구하고 여러 해에 걸쳐 계속 학생들이 개인적으로 찾아와서 묻는 일이 잦아 번거로워서 아예 글을 쓰는 겁니다.

### 1. 학부 졸업논문의 성격

여러분들이 쓰는 것은 ‘학부’ 졸업논문이라는 점을 먼저 강조하고 싶습니다. ‘대학원’ 졸업논문이 아니라는 뜻입니다. 따라서 각 지도교수가 요구하는 논문의 수준은 ‘학부’ 수준이지 그 이상이 아닙니다.

짐작컨대 여러분들은 ‘논문’이라고 지칭되는 글을 쓴 경험이 없다 보니, 그 명칭 자체에 부담을 느끼는 것 같습니다. 뭔가 새로운 것을, 뭔가 독창적인 아이디어를 담고 있는, ‘대단한’ 내지는 적어도 (여러분이 써보지 않았던) ‘특별한’ 것을 작성해야 할 것 같다고 느끼는 것이지요.

결론을 먼저 말씀드리자면 “전혀 그렇지 않다”입니다. 대학원의 졸업논문 중에도 독창적인 아이디어를 담고 있는 ‘대단한’ 논문은 그리 많지 않습니다. 하물며 학부 졸업논문에 이르러서야 말할 필요도 없겠지요.

사실 이러한 오해(?)는 ‘논문’이라는 명칭 자체에 대한 오해로부터 기인한다고 봅니다. 논문에 목적에 따른 여러 종류가 있습니다. 비록 독창적인 아이디어를 포함하고 있지는 않지만, 특정 분야나 주제에 대해 좋은 개관(overview & perspective)을 제공하는 논문도 있습니다. 또한 특별한 주제에 대해 독자들의 이해를 깊게 할 목적으로 그것을 강의하는 논문도 있습니다. 전자를 ‘개괄 논문(survey paper)’, 후자를 ‘튜토리얼 논문(tutorial paper)’이라고 부릅니다. 본격적으로 자신의 독창적인 아이디어를 개진하는 ‘연구 논문(research paper)’이 아니라도 다른 형식의 논문이 가능하다는 것이지요.

따라서 여러분은 ‘학부’ 졸업논문을 “(기존에 확립된 논문의) 형식을 갖춘, 잘 정리된 리포

트” 정도로 생각하면 되겠습니다. (여기서 ‘기존에 확립된 논문의 형식’이 무엇인지는 아래에서 추가로 설명하겠습니다.) 또 앞에서 나열한 논문의 종류들 중에서 맞는 것을 찾자면, 여러분이 쓸 논문은 ‘거의 모두’ 개괄 논문에 해당할 가능성이 높겠지요.

## 2. 논문의 주제

지금쯤은 여러분들이 다 알고 있으리라고 생각합니다만, 졸업논문의 주제는 자유롭습니다. 아무런 제한이 없어요.

여러분은 컴퓨터 및 관련분야를 전공으로 하는 학과에서 이미 3년 이상 적(籍)을 두고 있었 습니다. 그동안 많은 전공과목을 배우고 또 많은 이야기를 들었습니다. 그 많은 이야기들 중, 관심이 가는(절대적으로든 상대적으로든) 이야기가 하나쯤은 있지 않았나요? (하나도 없었다 고요? @\_@ 그렇다면 좀 문제네요... 그래도 상대적으로 ‘덜 싫었던 것’을 생각해 봅시다. 수강 한 과목의 담당교수나 교재로부터 듣거나 본 것이 자연스럽고 바람직하다고 생각하나, 꼭 그래 야만 한다고 강요하는 것은 아닙니다.) 바로 그런 것을 주제로 하면 되는 겁니다.

주전공이 따로 있고 컴퓨터 쪽을 복수전공하는 경우, 주제를 컴퓨터 과학 쪽에 한정하지 않 고 보다 유연하게 넓힐 여지도 있습니다. 예전의 예를 보면, 의류학과에서 컴퓨터를 복수전공 으로는 하는 학생은 아바타(avatar)에 대한 것을 주제로 하겠다는 경우도 있었고, 법학을 전공으 로 하는 학생이 디지털 시대의 지적소유권 문제에 대한 것을 주제로 삼겠다는 경우도 있었습 니다.

그렇지만 한 가지 여러분이 꼭 명심해야 할 사실이 있습니다. 아무리 개괄 논문이라고 할지 라도, 여러분이 작성하는 것은 **‘컴퓨터를 (주/복수)전공으로 하는 4년제 대학생의 졸업논문’**이 라는 점입니다. 따라서 그 논문을 ‘컴퓨터를 부전공이나 비전공으로 하는 4년제 대학생이 작성 할 수 있는 리포트’ 수준으로 만들면 곤란합니다. 그런 부/비전공 학생들이 이해하고 적을 수 있는 수준을 넘어서는, **‘컴퓨터나 정보기술에 관련된 기술적 내용을 포함한 리포트’**가 되어야 의미가 있을 것입니다.

보다 구체적으로 졸업논문의 주제가 될 만한 예시를 드는 일은 맨 뒤의 부록에서 하겠습니 다.

## 3. 피해야 될 주제

바로 앞부분 얘기는 중요하니 조금 더 부연해서 설명하겠습니다. 제가 과거 졸업논문을 지도 한 학생들과의 경험으로부터 예를 들어보죠. 어떤 학생이 「우리나라 u-City의 현황과 과제」 라는 제목으로 졸업논문을 쓰겠다고 합니다. 저는 이 주제가 컴퓨터정보학부 학생들이 쓰기에 매우 좋지 않다고 생각합니다. 이유는 다음과 같습니다.

- 주제가 너무 광범위하고 포괄적입니다. 아무리 대가(大家)라 하더라도 「컴퓨터 과학에 대하여」 라는 제목으로 논문을 작성하지는 않습니다. 아니, ‘컴퓨터 과학’은 고사하고 그 세부 분야 중 하나를 통째로 논문의 주제로 삼지는 않습니다. 「프로그래밍 언어에 대하여」 또는 「데이터베이스에 대한 고찰」 등의 제목으로 제대로 된 논문을 쓰는 일은 정상적인 상황에서는 대가에게도 불가능에 가까운 일입니다.
- 상기 주제(「우리나라 u-City의 현황과 과제」)의 핵심은 컴퓨터 과학보다는 행정학, 경영학, 보건학 등 다른 학문 분야에 속하는 것입니다. 따라서 컴퓨터 과학이나 정보기술을 주/복수전공으로 하는 학생은 문외한이라 좋은 글을 작성하기 어려운 기본적 제약이 따릅니다.
- 상기 주제에서 상대적으로 컴퓨터나 정보기술에 가까운 분야를 찾는다고 하더라도, 그 기술은 우리 학생들에게 그리 익숙하지 않은 전자공학(통신이나 신호처리 등)의 내용을 알아야 의미 있는 정도의 깊이까지 다가갈 수 있습니다. 역시 컴퓨터 과학, 그 중에서도 소프트웨어를 중심으로 공부한 여러분들이 접근하기에는 한계가 있습니다.
- 이렇게 상기 주제 중 그나마 있는 기술적 부분에 접근하는 데조차 제약이 따르다 보니, 이 주제로 작성한 글은 알팍한 백과사전식의 사실과 지식을 나열하는 모양새가 되기 십상입니다. 그런 사실과 지식은 알아도 그만 몰라도 그만이며, 컴퓨터를 전공으로 하지 않는 사람도 누구나 들으면 ‘아, 그런가 보다’ 하고 알아듣는 평이(平易)한 ‘그냥 이야기’에 불과합니다. 그걸 졸업논문으로 쓸 의미가 없는 거죠.

제가 위에서 말하고자 하는 바를 이해했다면, 이와 유사한 주제를 여럿 예로 떠올릴 수 있을 테고 그것들이 졸업논문의 주제로 적당하지 않다는 점을 쉽사리 깨달을 수 있으리라 봅니다. 예컨대, 「유비쿼터스 컴퓨팅의 다양한 응용에 대한 고찰」 과 같은 제목의 논문은 쓰레기(?)가 될 가능성이 매우 높습니다. 역시 이 주제의 기술적 부분(통신이나 센서 기술 등)은 우리 학생들이 알지 못하는 분야이니, 논문의 대부분 내용은 신문이나 잡지의 알팍한 기사 같은 것이 되기 일쑤입니다. TV에서 하는 1시간짜리 『유비쿼터스 세상이 몰려온다』 는 제목의 과학 다큐멘터리 프로그램을 함께 시청한 초등학생과 60대 어르신이 그 프로그램을 시청함으로써 알게 된 내용보다 앞서 언급한 논문의 내용이 확실히 나은 점이 뭐가 있겠습니까?

또 여러분들 스스로 한 번 생각을 해 보기 바랍니다. 만약 컴퓨터와 전혀 무관한 학과(사학과나 정치외교학과 등)의 저학년생이 교양 과목으로 「재미있는 유비쿼터스 세상」 을 듣고 학기말 리포트(term paper)로 이것(「유비쿼터스 컴퓨팅의 다양한 응용에 대한 고찰」)과 동일한 주제의 글을 작성한다고 생각해 봅시다. 그 학생이 정말 열심히 수업을 듣고 자료를 조사하고 리포트 작성에 시간을 투입하여 다듬는다고 가정할 때, 과연 여러분이 작성한 이 주제의 졸업논문보다 꼭 못한 글이 될 거라고 생각합니까? 그 저학년생이나 여러분이나 어차피 이 주제의 핵심기술 부분에 대해서는 마찬가지로 모르니, 남은 것은 비기술적인 내용의 사실과 지식을 누가 얼마나 충실하게 조사하고 짜임새 있게 나열하느냐는 점만 남는 거죠. 당연히 비전공 저학년생이 주전공인 여러분의 졸업논문보다 더 나은 리포트를 쓸 수도 있는 거고요. 이걸 우리가 도저히 참아줄 수 있는 상황이 아니지 않습니까?

이것과 동일한 성격의 주제들은 얼마든지 나열할 수 있습니다. 「RFID의 응용 분야에 대한 조사」(상기 주제와 동일한 문제점), 「스마트폰 시장의 경향」(타 분야 전공자에게 보다 맞는 주제), 「차세대 모바일 플랫폼의 추세」(타 분야 전공자에게 보다 맞는 주제), 「유비쿼터스 환경의 보안 문제」(상기 주제와 동일한 문제점—통신이나 암호화 기술 등의 세부 내용을 모름) 등등.

#### 4. 주제의 구체화

큰 틀에서 관심이 가는 주제가 하나 정해졌다면, 일단 성긴 제목을 정할 수 있습니다. 「...의 현재 동향」, 「...의 역사와 전망」, 「...에 대하여」, 「...의 구현을 위한 기법 조사」와 유사한 모양을 갖춘 제목을 말합니다.

또 논문 제목은 여러분들이 써 가면서 바꿀 수도 있습니다. 지금 정한 것이 11월(혹은 후기 졸업의 경우 5월)까지 ‘꼭’ 그대로 유지되지 않아도 됩니다(그렇다고 11월 초에 최종 제목을 정하라는 뜻은 아닙니다).

앞에서 말한 것과 같은 포괄적이고 일반적인 제목을 가지고 시작한 뒤, 여러분들이 자료를 찾고 공부하면서 제목을 점점 더 구체적으로 만들 수 있겠지요. 예를 들어 처음에는 「컴퓨터 보안에 대하여」라고 시작하였다가, 「컴퓨터 내의 정보 보안에 대하여」 또는 「컴퓨터 네트워크 보안에 대하여」로 보다 구체화되고, 또 다시 「컴퓨터 내의 정보 보안을 위한 암호화 기법에 대하여」 또는 「컴퓨터 네트워크 보안을 위한 방화벽 기법에 대하여」 등으로 더욱 구체화 된다는 것이지요. 일반적으로 좋은 논문은 그 종류에 관계없이 구체적인 제목을 가지고 있습니다.

#### 5. 논문의 형식과 내용

이제 형식에 대해서 이야기를 하겠습니다. 논문뿐만 아니라 모든 논리적인 글이 그렇다시피 그 본문의 구성은 ‘서론’, ‘본론’, ‘결론’으로 되어야 합니다. 그리고 논문의 경우는 필히 ‘참고문헌’ 부분이 있어야 합니다. 여기서 ‘참고문헌’을 제외한 나머지 부분은 다양한 제목을 가질 수 있습니다. 예컨대 ‘서론’ 대신에 ‘들어가는 말’이나 ‘시작하는 말’, ‘서언’ 등이 가능하겠죠. ‘결론’도 마찬가지입니다. 또한 ‘서론’, ‘본론’, ‘결론’ 모두 두 개 이상의 장 또는 절로 세분화 될 수 있습니다.

‘참고문헌’에는 논문을 작성하면서 참고한 책, 논문, 기사, 웹 페이지 등을 참고문헌 표기 형식에 맞도록 적습니다. 이 ‘참고문헌’에 열거된 것들은 논문의 본문 어딘가에서 ‘꼭’ 참조가 되어야 합니다. 참조가 되지 않은 문헌이라면 ‘참고문헌’에 나열되어 있을 이유가 없는 것이지요. 논문을 처음 작성하는 사람이 빠지기 쉬운 함정들 중 하나가, 다른 사람의 말을 마치 자기가 처음 하는 것인 양 그 말의 출처를 밝히지 않는 것인데, 이것은 일종의 ‘사기(cheating)’일 뿐

만 아니라 자기 논문의 신뢰성과 무게를 오히려 떨어뜨리는 일입니다.

논문이 담고 있어야 할 일반적 내용에 대해서도 간단하게 언급하겠습니다.

‘서론’에는 왜 이러한 주제를 다루는지에 대한 서술이 있어야 합니다. 이 주제로 논문을 쓰게 된 동기(motivation)이죠. 왜 이 주제가 가치가 있는지를 말할 수 있다면 더욱 좋겠죠. 그리고 ‘서론’의 끝 부분에는 논문의 구성에 대해 서술해야 합니다. 이러이러한 순서로 이야기를 진행하겠다는 것이지요.

‘본론’에 해당하는 부분에서는 여러분들이 자료를 모으면서 공부한 내용을 적습니다.

‘결론’ 부분에서는 논문의 앞에서 논의한 것을 간단히 정리하는 부분이 있어야 합니다. 이 논문에서 결국 무엇을 말한 것인지 적는 것이죠. 그리고 마지막으로 자신이 이 논문을 쓰면서 느낀 점이나 논문의 주제와 관련한 미래에 대한 전망 등을 쓰면 아주 좋죠.

## 6. 졸업논문을 쓰는 목적과 여러분이 얻어야 할 것

모든 일이 그렇다시피 졸업논문을 작성하는 목적에 대해 제대로 인지해야 이 일에 대한 자기-동기(self-motivation)를 부여할 수 있을 테고, 그래야만 제대로 일도 할 수 있을 것입니다.

제가 생각하기에(저만 그렇게 생각하는 것도 아닙니다만) 우리 학부 학생들이 취업하고자 할 때 가장 부족하다고 여겨지는 스펙이 스스로 만든 ‘프로그램의 포트폴리오(portfolio)’가 갖추어져 있지 않은 것입니다(누가 “학생이 만들어 본 프로그램에 어떤 것이 있죠?”라고 물으면 대답할 만한 게 없다는 뜻입니다. 버블 정렬이나 이진 검색을 짜봤다고 대답할 수는 없으니까요). 컴퓨터 관련 학과 학생으로서는 어쩌면 가장 중요할 수도 있는 스펙이 결여되어 있는 셈이죠.

이것은 자칫 학생이 지원 단계나 인터뷰 단계 등에서 스스로 자신감을 잃고 주눅 들게 하는 결과를 가져오기도 합니다. 취업에 성공하기 위한 가장 중요한 요인은 적극적인 자세(attitude)라고 볼 수 있는데, 지원자 스스로 주눅이 들면 결과는 보나마나죠.

따라서 **졸업논문보다는 졸업작품을 선택하기를 모든 교수들이 강력히 권장**하며 그러기 위해 저학년 때부터 프로그래밍을 대하는 태도를 가다듬기를 바랍니다. 졸업논문은 여러 사정으로 불가피할 때 선택할 차선(次善)인 셈인 거죠.

자, 어쨌든 이 차선인 졸업논문을 선택했다고 생각합시다. 그리고 여러분이 취업 인터뷰 자리에서 지원자로서 면접관과 대화를 나누는 장면을 상정해 보죠.

면접관: “아무개 씨, 학교 다니면서 만들어 본 프로그램은 어떤 것들인가요?”

지원자: “... 수업 시간에 과제로 짠 것들 이외에 특별히 언급할 만한 것은 없는 것 같습니다.”

면접관: “아니, 컴퓨터 전공 학과인데 졸업할 때 프로그램 작품 같은 것 안 하나요?”

지원자: “졸업요건이 작품을 할 수도 있고 자격증과 논문을 할 수도 있는데, 저는 논문을 썼습니다.”

면접관: “그렇군요. 그럼 아무개 씨는 뒤에 대해서 논문을 쓰셨나요?”

지원자: “... RFID ... 유비쿼터스가 어찌구 ... 차세대 모바일 환경이 저찌구 ... 보안이 더욱 중요해질 것 ... 뭐 이런 주제로 논문을 썼습니다.”

면접관: “지금 말한 주제는 그 기반 기술이 전자, 통신, 센서, 수학 등과 밀접하게 관련이 있거나 아니면 기술보다는 정책과 관련이 깊은데, 아무개 씨의 성적증명서를 보면 학과의 교과과정이 주로 컴퓨터 소프트웨어 중심으로 구성되어 있는 듯합니다. 논문 작성시 기술적 부분을 다루는 게 어렵지 않았나요?”

지원자: “... 그래서 논문을 기술적 내용보다 적용되는 사례를 살펴보는 식으로 작성했습니다.”

면접관: “그렇게 되면 논문이라는 게 ‘좋은 게 좋은 거다’라거나 ‘이런 것도 있고 저런 것도 되니 좋다’는 식의 뻘한 말만 담고 있게 되지 않나요? 그런 논문을 써도 졸업이 되나요?”

지원자: “... 솔직히 말씀드리면, 말이 논문 심사지 매우 형식적이라서 아무 것이나 제출만 하면 그냥 통과되는 분위기였습니다.”

면접관: “...”

어때요? 이쯤 되면 여러분이 면접관이라도 이런 지원자를 뽑고 싶겠습니까? 그렇기 때문에 제가 앞의 3절에서 논문의 주제로서 피해야 될 것을 강조하여 얘기한 것입니다. 여러분이 비록 프로그램의 포트폴리오는 갖추지 못 했지만, 졸업논문 작성을 여러분의 취업이나 미래 커리어 구축을 위해 효과적으로 사용해야 되지 않겠습니까?

그러기 위해서는 여러분이 작성한 졸업논문의 주제가 면접관의 흥미를 끌거나, 비록 범위가 좁기는 하나 그만큼 나름 깊이가 있게 공부를 하여 면접관에게 뭔가를 가르쳐 줄 수 있어야 합니다. 면접관이 여러분보다는 훨씬 공력(功力)이 깊을 테지만, 총론에서는 매우 강하나 모든 분야의 각론에 다 강한 것은 아니니까 여러분이 적절한 주제를 선정하고 성실하게 준비하여 작성하면 충분히 가능한 일입니다. 일단 면접관에게 면접자가 뭔가를 가르쳐 주는 상황이 되면 그건 면접자에게 매우 유리합니다.

## 7. 맺는 말

마지막으로, 여러분이 제발 그러지 말기를 제가 바라는 것에 대해 이야기하겠습니다.

제가 제일 바라는 것은, 여러분이 이 논문을 쓰는 과정을 통해 투입한 자원(시간과 마음 씀, 종이와 잉크 값, 전기료 등)이 허비되는 것이 아니게 하라는 것입니다.

여러분이 논문을 쓰면서 그 논문이 다루고 있고 여러분이 관심을 가지고 있던 주제에 대해 뭔가 배우게 되었다면, 소비한 자원은 허비된 것이 아니겠지요. 여러분이 논문의 구조를 구상하고 각 부분에서 기술해야 할 작은 주제와 내용을 배분해 보면서 기술적인 글을 어떻게 작성해야 하는지에 대해 좀 더 잘 알게 되었다면, 소비한 자원은 크게 보상을 받는 것이겠지요.

그런데 그게 아니라 걱정만 하면서, 실제 몸으로 움직여 자료를 찾고 공부하는 일은 하지 않다가, 막판에 가서 다른 사람의 ‘전적인’ 도움으로 논문을 만들거나 웹에서 찾은 문서를 그대로 찍어서 만든 논문을 제출하고 끝난다면, 그 동안 소비한 자원은 그냥 허비되는 것이 되겠지요. 그리고 그러겠지요. “도대체 이런 일을 왜 시키는 거야!”

맞습니다. 그런 일은 전혀 할 가치가 없는 것 맞습니다, 맞고요. 그런데 그런 경우 그 일이 전혀 할 가치가 없게 만든 사람이 바로 여러분 자신이라는 것을 인정해야 합니다. 그 일이 가치가 있게 만드는 사람도 또한 여러분 자신입니다.

여러분이 배우고 싶은 주제를 가지고 이번 여름방학(내년 여름에 졸업예정인 학생의 경우는 이번 겨울방학)에 두어 주 바짝 공부하면 아주 좋은 졸업논문을 쓰게 될 겁니다. 배우는 것도 많을 것이고요. 파이팅!

## 부록: 바람직하다고 여겨지는 논문 주제의 예시

아래에 여러분이 수강한 각 전공과목 별로 졸업논문의 주제로 생각해 볼만한 것들을 몇 가지 나열해 보겠습니다. 너무 당연한 얘기지만 여기에 열거된 것들은 예시일 뿐이며, 여러분들이 제가 앞에서 말한 것을 이해했다면 더 많은 주제들을 생각해낼 수 있을 것입니다.

### (1) 프로그래밍언어 분야

- C 언어를 사용한 객체-지향 프로그래밍: 알다시피 C 언어는 명령형 프로그래밍 언어이고, 객체-지향 프로그래밍 언어가 필수적으로 갖추어야 할 ‘추상 데이터 타입(ADT)’, ‘상속’, ‘다형성’ 등의 기능을 제공하지 않습니다. 그러나 C 언어의 다양한 프로그래밍 기법을 이용하여 어설프게나마 객체-지향 프로그래밍과 유사하게 프로그램을 작성할 수 있습니다. 어떻게 하면 될까요? 이것을 공부하면 여러분의 C 프로그래밍과 객체-지향에 대한 이해가 깊어질 것입니다.
- 객체-지향 언어들의 다중상속 비교: 다중상속(multiple inheritance)은 객체-지향 프로그래밍 언어에서 ‘뜨거운 감자’와 같은 주제입니다. 필요할 때가 분명히 있음에도 불구하고, 그것의 도입으로 인해 야기되는 여러 가지 문제점들이 있기 때문입니다. 각 객체-지향 언어들마다 다중상속 문제에 대응하는 방식은 차이가 있으며, 그에 대한 논리도 있습니다. 몇 가지 주요 객체-지향 언어들의 다중상속 방식을 꼼꼼히 비교해 보는 일은 여러분의 객체-지향에 대한 이해를 심화시키는 효과를 가져 올 것입니다.
- 프로그래밍 언어의 정형적 의미론 중 공리적 의미론 고찰: 보다 중요한 응용 분야에 컴퓨터가 사용됨에 따라 신뢰성(reliability) 높은 소프트웨어 제작에 대한 요구는 점점 더 높아지고 있습니다. 신뢰성 높은 소프트웨어, 즉 버그(bug) 없이 명세(specification)와 일치하는 소프트웨어를 작성하거나 검증하기 위해서는 구현 언어의 의미론(semantics)이 명확하게 정의되어 있어야 합니다. 이런 뜻에서 프로그래밍 언어의 정형적 의미론(formal se-

antics)은 여전히 관심을 받고 있습니다. 주요 정형적 의미론 중 하나를 학부생 수준에서 공부해 보는 것은 의미 있는 일이 될 것입니다.

## (2) 알고리즘/계산이론 분야

- 이진 탐색(binary search)의 분할 위치에 따른 성능 비교: 일반적인 이진 탐색은 탐색해야 할 군집 데이터의 정 가운데 항목과 찾으려고 하는 키(key)를 비교합니다. 그 비교의 결과 항목을 찾거나, 아니면 앞의 반 또는 뒤의 반을 탐색 공간에서 제외시키게 되죠. 만약 군집 데이터의 정 가운데가 아닌 위치(예컨대 앞에서  $\frac{1}{3}$  위치)의 항목과 키를 비교하면 어떨까요? 그 비교의 결과 항목을 찾거나, 아니면 앞의  $\frac{1}{3}$  또는 뒤의  $\frac{2}{3}$ 를 탐색 공간에서 제외시키게 되겠죠. 어떤 게 효율적일까요? 무작위하게 만든 샘플 데이터에 대해 여러 번 수행시켜 평균치를 구하면 의미 있는 결과를 구할 수 있을 겁니다.
- P와 NP 문제의 역사와 정의: 보통 학부 과정에서는 ‘계산이론’이나 ‘형식언어’, ‘오토마타’같은 과목이 개설되더라도 P와 NP 문제를 깊이 다루지는 않습니다. P와 NP의 관계는 현재 이론 컴퓨터 과학에서 가장 중요한 문제입니다. 이 주제에 대해 그 역사적 배경을 살펴보고 지금 도달해 있는 상태를 요약해 보는 일은 재미도 있고 차별화시켜 다른 사람에게 설명해 주기 좋은 내용을 여러분이 얻게 되는 것입니다.
- ‘여행하는 세일즈맨 문제(Traveling Salesperson Problem; TSP)’에 대한 다양한 해법: TSP 문제는 오래된 대표적 NP 문제입니다. 이 문제에 대한 다양한 접근법이 존재합니다. 정통적인 알고리즘적 기법(탐욕적 방법이나 분기한정법 등)뿐만 아니라 인공지능 기법을 이용한 접근방법도 있습니다. 따라서 이 단일 문제만으로 한정하더라도 배우는 게 제법 있을 것입니다.
- ‘최소 신장 트리(Minimum Spanning Tree)’ 알고리즘들의 최적화를 위한 자료구조: Prim이나 Kruskal 알고리즘의 효율성을 조금 더 높이기 위해 특이한 자료구조를 사용하는 방법이 있습니다. 알고리즘 수업을 들은 학생들은 해당 절의 마지막 부분에 참고문헌을 적시하며 간단하게 언급하는 것을 본 기억이 어찌면 있을 지도 모르겠습니다. 최소 신장 트리뿐만 아니라 다른 문제의 경우에도 비슷한 접근방법이 공통적으로 나타납니다. 그 중 한 문제를 골라서 최근까지의 성과를 살펴보는 것입니다.

## (3) 데이터베이스 분야

- 다양한 데이터베이스 정규형(Normal Form)과 그 의미: 여러분은 데이터베이스 시간에 여러 가지 데이터베이스 정규형(제1 정규형, 제2 정규형, Boyce-Codd 정규형 등)을 배웠을 것입니다. 아마 학부 수업시간에는 지금까지 등장한 모든 데이터베이스 정규형을 다루지 않았을 것이라고 짐작합니다. 수업시간에 다루지 않은 정규형들을 포함하여, 정규형을 정의하고 그 목적과 예제, 장단점 비교, 실제 구현에서의 문제점 등을 살펴보는 것입니다.

#### (4) 운영체제 분야

- 실험을 통한 페이지 대체 알고리즘의 효율성 비교: 여러분은 수업 시간에 여러 가지 페이지 대체(page replacement) 알고리즘을 학습했습니다. 예를 들어, FIFO(First-In First-Out)이나 LRU(Least Recently Used)같은 것 말이죠. 또한 LRU가 효율적이라는 결론만 들었을 가능성이 높습니다. 그리 복잡하지 않은 프로그램으로 과연 LRU가 정말 효율적인지를 시뮬레이션을 통해 확인할 수 있습니다. 이 프로그램의 완성도를 조금 높이면 졸업논문의 부수 작업이 아니라 독자적인 졸업작품으로 사용할 수도 있다고 봅니다.

\*\*\*\*\* 논문 주제는 계속해서 추가될 수 있습니다 \*\*\*\*\*