APPENDIX I
PROOF OF THEOREM 1

In this section we present the proof of theorem 1.

Theorem 1: Let E_1 and E_2 be the set of solutions to Sys1 and Sys2 below.

$$\max_{C1,C3,C5-C8,C10}(4) \tag{Sys1}$$

$$\max_{C1-C11}(4) \tag{Sys2}$$

We have

$$E1 = E2. \tag{Th}$$

Proof: We say that $v \in E_1$ is a solution that satisfies Sys1 and we write $v \vdash Sys1$. We first prove that all solutions of Sys2 are solutions to Sys1 ($E_1 \subseteq E_2$). Then we prove the other direction ($E_2 \subseteq E_1$).

A. *Left:* $E_1 \subseteq E_2$

We claim

$$\forall v \in E_1 . v \vdash C2, C4, C9. \tag{app.1}$$

We rewrite app.1 as

$$\neg\exists v \in E_1 . (v \nvdash C2) \vee (v \nvdash C4) \vee (v \nvdash C9), \tag{Th'}$$

and we show Th' case by case.

Case: $\neg\exists v \in E_1 . v \nvdash C2$**.** Recall that C2 ensures that projects that are not selected in a quarter, have no resources allocated to them during that quarter. Assume $v \nvdash$C2, thus v is configured such that

$$\exists p, t . \boldsymbol{S}^v[p][t] = 0 \ \wedge \ \exists k . \boldsymbol{A}^v[t][p][k] > 0. \tag{app.2}$$

We construct the solution u exactly similar to v except for the following.

$$\forall p, t . \boldsymbol{S}^u[p][t] = 0 \ \wedge \ \forall k . \boldsymbol{A}^u[t][p][k] = 0. \tag{app.3}$$

We have $u \vdash C1, C3, C5 - C8, C10$. We also have $(4)^u > (4)^v$ since with u we reduce effort without increasing cost or reducing revenue. Thus $v \notin E_1$.

Case: $\neg\exists v \in E_1.v \nvdash C4$**.** Recall that C4 guarantees that we do not work on projects that are not selected. Assume $v \nvdash$C4, thus v is configured such that

$$\exists p.\boldsymbol{\gamma}^v[p] = 0 \ \wedge \ \sum_{t=1}^{T} \boldsymbol{S}^v[p][t] > 0 \tag{app.4}$$

This is equivalent to

$$\exists p, (\boldsymbol{\gamma}^v[p] = 0) \wedge (\exists t, \boldsymbol{S}^v[p][t] = 1) \tag{app.5}$$

We substitute for $\boldsymbol{S}^v$ with its implications using C1 to obtain

$$\begin{aligned} \exists p, (\boldsymbol{\gamma}^v[p] = 0) \wedge (\exists t, \forall j, \boldsymbol{B}^v[t][p][j] \geq \boldsymbol{W}^v[p][j]) \\ \wedge (\exists j, \boldsymbol{W}^v[p][j] > 0). \end{aligned} \tag{app.6}$$

We construct u that is exactly similar to v except for the following.

$$\forall p.\boldsymbol{\gamma}^v[p] = 0 \ \Rightarrow \forall t, j.\boldsymbol{B}^u[t][p][j] = 0. \tag{app.7}$$

We have $u \vdash C1, C3, C5 - C8, C10$. We also have $(4)^u > (4)^v$ since with u we reduce effort without increasing cost or reducing revenue. Thus $v \notin E_1$.

Case: $\neg\exists v \in E_1.v \nvdash C10$**.** Recall that C9 guarantees that if we are busy in one quarter, then there is at least one project selected in that quarter. Assume $v \nvdash C9$, thus v is configured such that

$$\exists t, (\boldsymbol{b}^v[t] = 1) \wedge (\forall p, \boldsymbol{S}^v[p][t] = 0). \tag{app.8}$$

We construct u that is exactly similar to v except for the following.

$$\forall p, t.\boldsymbol{S}^u[p][t] = 0 \Rightarrow \boldsymbol{b}^u[t] = 0. \tag{app.9}$$

We have $u \vdash C1, C3, C5 - C8, C10$. We also have $(4)^u > (4)^v$ since with u we reduce cost without increasing effort or reducing revenue. Thus $v \notin E_1$. We have proved all cases of Th' and thus $E_1 \subseteq E_2$.

B. Right: $E_2 \subseteq E_1$

All solutions of Sys2 satisfy all constraints of Sys1 as they satisfy more constraints. We need to prove that they actually maximize (4). The three cases of $E_1 \subseteq E_2$ show that satisfying the constraints C2, C4, and C9 after satisfying all other constraints, reduces cost or effort monotonically without decreasing revenue. This concludes our proof. ■

APPENDIX II
IMPORTANCE OF SKILLS

In this section we present a way to rank the importance of software skills according to two indices: The $neededIndex$ and the $availableIndex$.

A. $neededIndex$ and $availableIndex$

The strength that a project requires for a certain skill j indicates how much skill j is needed by project p. We define $neededIndex[j]$ to be an indicator of how much skill j is needed by the set of available projects. $neededIndex[j]$ is the normalized summation of the required strengths in j for all the available projects over a scale of 0 to H. Similarly, we define $availableIndex[j]$ to be an indicator of how much skill j is available in a certain workforce. $availableIndex[j]$ is the normalized summation of the available strengths in j for all the developers over a scale of 0 to H. Comparing $neededIndex[j]$ and $availableIndex[j]$ allows to determine the match or mismatch between how much skill j is needed and how much it is available in a certain country.

B. Lebanese Case

We consider the 11 projects and the 11 companies that we surveyed, then we compute $neededIndex$ and $availableIndex$ for all the 197 skills. Figure 1 shows that the most needed skills are the general programming and algorithmic skills and the web development ones. While the least significant skills are the artificial intelligence (AI), the mathematical, and the electrical engineering skills. For the 20 most needed skills, we can see that there is a good match between the $neededIndex$ and the $availableIndex$. This shows that the academic emphasis on these skills goes well with the industry structure and requirements. On the other hand, if we consider the 20 least significant skills, we can see that the $availableIndex$ is much higher than the $neededIndex$ for all of the skills. This shows that the emphasis on math courses, AI and general electrical engineering concepts is not being valued by the Lebanese software industry because these skills do not match with the spectrum of projects that the Lebanese industry considers.

Our results agree with the findings of Lethbridge [19]. We both conclude that universities are giving more emphasis on math skills than what the developers actually think to need at work. Similarly, our results agree on the importance of general programming skills in the current software development market. The electrical engineering concepts justifiably belong to the least

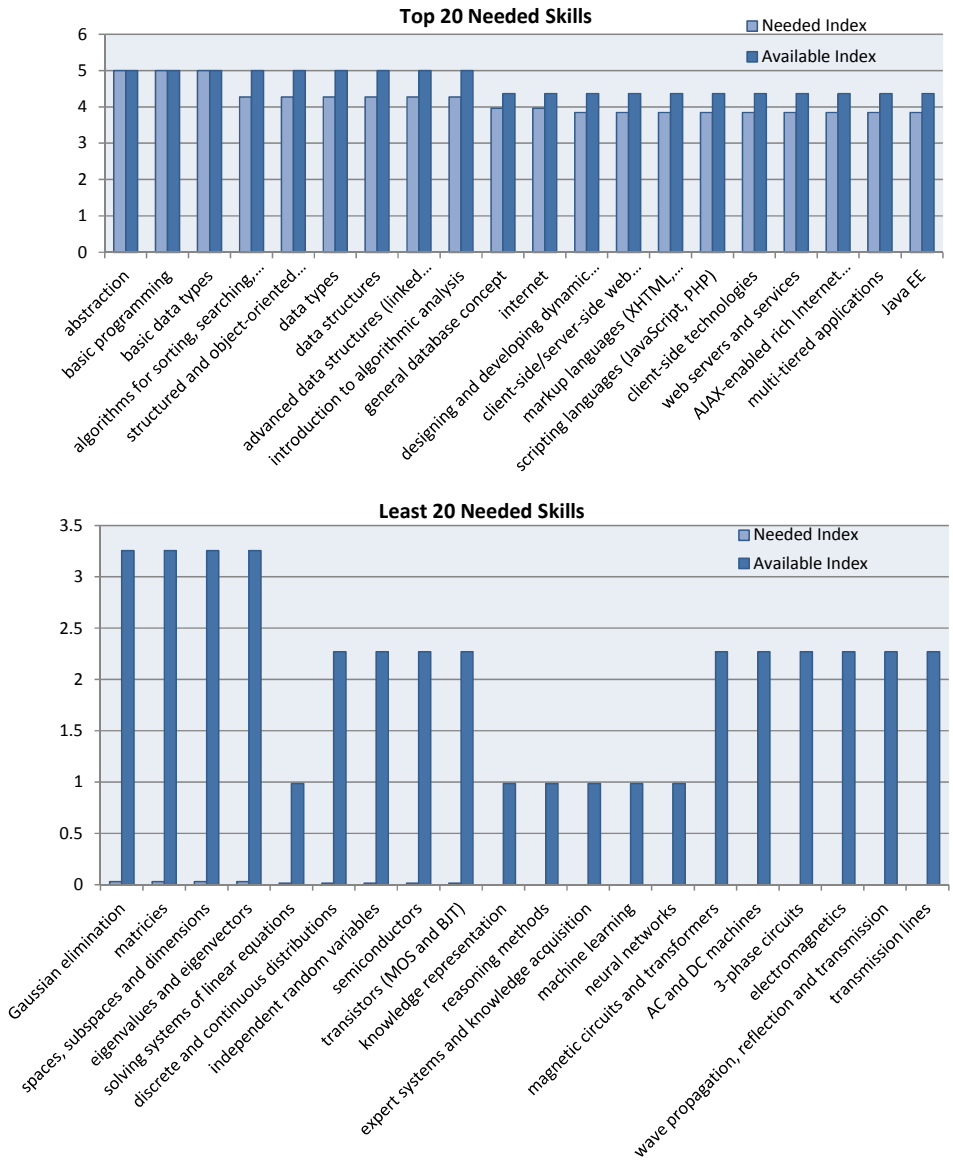

Fig. 1. Top 20 Needed Skills and Least 20 Needed Skills

important skills. But since universities in Lebanon do not generally offer a software engineering major and offer instead Computer Engineering or Electrical Engineering, these skills are being reinforced in an unneeded manner for graduates that decide to pursue a software track.

APPENDIX III
EXHAUSTIVE LIST OF SOFTWARE SKILLS

In table I we present an exhaustive list of 197 software skills that are taught in Lebanese academia. We group them into 37 categories.

Categories	**Skills**
1. Introduction to Programming	1- abstraction 2- basic programming 3- basic data types
2. Data Structures and Algorithms	4- algorithms for sorting, searching, and indexing 5- structured and object-oriented programming 6- data types 7- data structures 8- advanced data structures (linked lists, binary trees, heaps, B-trees, graphs) 9- introduction to algorithmic analysis
3. Design and Analysis of Algorithms	10- design and analysis of efficient algorithms 11- median and order statistics algorithms 12- divide-and-conquer design strategy 13- polynomial and matrix multiplication algorithms 14- number-theoretic algorithms 15- dynamic programming and greedy algorithms 16- graph algorithms (graph traversal algorithms) 17- minimum spanning tree, shortest path algorithms 18- NP-completeness and intractability 19- algorithms (advanced searching, sorting, selection, graph and matrix) 20- complexity consideration 21- consolidating algorithm design and programming techniques
4. Operating Systems	22- processes and threads 23- concurrency and synchronization 24- scheduling and resource management 25- file systems 26- IO devices 27- virtual memory 28- parallel and distributed systems 29- operating systems security and protections 30- network structures 31- Contemporary operating systems (UNIX)

5. Database Systems	32- data modeling (entity relationship models) 33- relational algebra and calculus 34- integrity constraints 35- file organization and index files 36- SQL 37- relational DBMS 38- query processing and optimization 39- transaction processing 40- concurrency control in DBMS 41- database recovery 42- database security 43- web deployed database systems
6. Programming Language Design and Implementation	44- programming paradigms 45- object-oriented (Java, C++ and C#) 46- functional (Haskell) 47- logic (Prolog) 48- Compiler construction, virtual machines, intermediate languages, concurrency
7. Software Engineering	49- requirement elicitation and specification 50- software analysis and design 51- design verification and testing 52- generic data models 53- management of software teams 54- software testing and validation 55- system documentation 56- system implementation
8. Advanced Algorithms and Data Structures	57- algorithms (advanced searching, sorting, selection, graph and matrix) 58- analyzing the complexity of algorithms 59- designing efficient computer algorithms 60- proving correctness of algorithms
9. Theory of Computation	61- regular expressions 62- finite and pushdown automata 63- context-free grammars and parsing 64- Turing machines 65- complexity theory
10. Web Programming and Design	66- designing and developing dynamic web-based applications 67- client-side/server-side web programming 68- markup languages (XHTML, Dynamic HTML and XML)

	69- scripting languages (JavaScript, PHP)
	70- client-side technologies
	71- web servers and services
	72- AJAX-enabled rich Internet applications
	73- multi-tiered applications
	74- Java EE
11. Human Computer Interaction	75- psychological principles of human-computer interaction
	76- design of user interfaces (windows, menus,commands,Voice and natural language I/O)
	77- user interface architectures and APIs
	78- prototyping
	79- evaluating user interfaces
	80- Internationalization and localization of user interfaces
12. Computer Graphics	81- interactive graphics programming
	82- graphics systems
	83- 2D and 3D graphics
	84- OpenGL
13. Computer-aided Geometric Design	85- modeling objects
	86- curve approximation and interpolation
	87- surface approximation and interpolation
14. Artificial Intelligence	88- knowledge representation
	89- reasoning methods
	90- expert systems and knowledge acquisition
	91- machine learning
	92- neural networks
15. Multimedia Programming	93- scripting language
	94- multimedia products
16. Network Programming	95- sockets
	96- low level networking programming
	97- networking applications (client/server, peer-to-peer)
17. Compiler Construction	98- parsing theory and parser generators
	99- syntax-directed code generation
	100- dynamic storage allocation
	101- code optimization
	102- dataflow analysis
	103- register allocation
	104- logic of programming
18. Multimedia Design	105- computer-enhanced and computer generated environments
	106- designing multimedia products
19. Foundations of Digital Media	107- digitizing media (images, audio, video)

	108- manipulating digital media
	109- media compression
	110- media generation
20. Digital Imaging	111- imaging (pixel-based and vector-based)
	112- digital image creation and manipulation
	113- computer graphics color, textures, lighting and shading
	114- 3D objects modeling
21. Numerical Analysis Techniques	115- numerical analysis techniques
	116- number representation
	117- Monte-Carlo methods
22. Communications Systems	118- analog signals (transmission and reception)
	119- analog communication systems with noise
	120- analog to digital conversion and pulse coded modulation
	121- digital signals (transmission and reception)
	122- digital communication systems with noise and inter-symbol interference
23. Computer Networks	123- data communications
	124- networks (WAN and LAN)
	125- packet switching and routing
	126- congestion control
	127- communications architecture and protocols
24. Computer Organization	128- computer systems organization
	129- assembly level programming
	130- designing components of a von Neumann computer system
25. Computer Architecture	131- digital logic and systems
	132- machine level representation of data
	133- machine organization
	134- memory system organization and architecture
	135- CPU implementation
	136- virtual machines
26. Signals and Systems	137- continuous and discrete-time signals and systems
	138- linear time-invariant systems
	139- stability analysis
	140- sampling of continuous-time signals
	141- z-transform
	142- discrete Fourier transform
	143- time and frequency domain representations of discrete-time signals and systems
27. Digital System Design	144- digital systems design concepts and representation
	145- synchronous digital systems

	146- standard logic (SSI, MSI)
	147- programmable logic (PLD, FPGA)
	148- finite state machine design (FSM)
	149- computer-aided design software in VHDL
28. Analog and Digital Circuits	150- circuits (frequency selective filters, Opamps ...)
	151- integrated circuits (digital and analog)
29. Electronics	152- semiconductors
	153- transistors (MOS and BJT)
	154- amplifiers and their frequency response
	155- oscillators
30. Electric Machines and Power Fundamentals	156- magnetic circuits and transformers
	157- AC and DC machines
	158- 3-phase circuits
31. Electromagnetics	159- electromagnetics Principles
	160- wave propagation, reflection and transmission
	161- transmission lines
32. Electric Circuits	162- circuit analysis
	163- Laplace transform and Fourier transform
	164- power relations and calculations
33. General Engineering Tools	165- SPICE
	166- LabView
	167- MATLAB
34. Basic Computer Software	168- wordprocessing
	169- spreadsheets
	170- general database concept
	171- internet
	172- Visual Basic
35. Differential Equations	173- linear differential equations
	174- series solutions
	175- Bessels and Legendres functions
	176- Laplace transform
36. Discrete Structures	177- Logical reasoning
	178- sets, relations and functions
	179- mathematical induction
	180- counting and enumeration methods
	181- probability theory
	182- algorithm analysis and complexity
	183- truth tables
	184- graphs, trees, strings and languages

37. Linear Algebra	185- systems of linear equations 186- Gaussian elimination 187- matricies 188- spaces, subspaces and dimensions 189- eigenvalues and eigenvectors 190- numerically solving systems of linear equations
38. Calculus and Analytic Geometry	191- sequences and series 192- coordinates (cylindrical and spherical) 193- multivariable functions 194- partial derivatives 195- multiple integrals
39. Introduction to Probability and Random Variables	196- discrete and continuous distributions 197- independent random variables

TABLE I: Table of Skills