Constructing Socio-Technical Patches for Information Foraging:
A Requirements Traceability Questions Case Study

I. Introduction

i. Concept: What is Information Foraging? (IF)
ii. Work in IF currently assumes patches
iii. Consider Jira & Requirements Traceability Questions (RTQs)
  1. Concept: What is RTQ?
  2. patches aren't given in RTQ
iv. Thesis: In this paper, (problem) we'll demonstrate the construction of patches in contexts where patches aren't evident by (solution) examining the properties of a socio-technical information environment

b. Transition: ???

II. Background & Related Work

i. Concept: Spreading Activation (PFIS); Activation == Scent
ii. Concept: artifact-based requirements traceability
  1. Codebook
  2. (ICSE2018, Section 2.2)
iii. ???

b. Transition: now that we're familiar with heterogeneous graphs (via Codebook), let's use that tool to build a network out of our Jira environment and start building patches within that network

III. Examining Requirements Socio-Technical Graphs

a. Information Environment (Intro to Jira) (ICSE2018, Section 3)

i. Potential Figure: Table of Jira Projects
ii. Concept: Issue/Assignee/Creator/Comment/Commenter/Reference as nodes
iii. Why were projects chosen?
iv. How are questions and answers identified?
v. Justification: Why do we need to look at this heterogeneously?
vi. More from (ICSE2018, Section 3) if we need to fill space

b. Constructing Requirements Socio-Technical Graphs (RSTG)

i. Potential Figure: Pseudocode of Graph Construction
ii. How are the nodes connected?

c. Information Foraging Properties of Requirements Socio-Technical Graphs

i. Figure: Pie Chart of the four classes
ii. Potential Figures: Examples of each classes Qs, As, and Graphs?
iii. Classes of foraging paths between Qs and As
  1. Referenced in Comment (1 degree)
  2. Issue Creator/Assignee (2 degrees)
  3. 3 to 4 Degrees of Socio-Technical Separation
    1. Relationship of Answerer to Creator/Assignee
    2. Relationship of Answerer to Asker
  4. Answerer not yet in network

d. Transition: We've built a graph and we understand the key information paths that should be traversed in a foraging task. What do we do to encode these paths into patches?

IV. Constructing Socio-Technical Patches for Information Foraging

a. Spreading Activation over Requirements Socio-Technical Graphs (SA-RSTG)

i. Figure: Spreading Activation Pseudocode

ii. Graph + Weight + Spreading Activation = Scent

iii. Design process that led to assignment of weight

1. Weight == Knowledgeable

iv. Design process that led to assignment of activation

v. Discussion of graph construction only before $t_{question}$

b. Delineating Patches within SA-RSTG

i. High Activation == High Scent, right? Therefore…

ii. Patch = Graph(Activation >= Cutoff)

1. What's the cutoff?

c. Transition: Does this reasoning produce useful patches?

V. Results

a. Qualitative Result: Example of SA-RSTG Patch Connecting Q to A

i. Figure: A graph with a question, answer, visible weights and activation, and delineated patch

ii. A discussion of how an asker might forage a SA-RSTG patch to find the answer that's visible in the diagram

b. Quantitative Results

i. Patch Size

ii. Answer Within Patch

iii. Also: How close is predator to prey when prey hasn't yet entered network?

c. Transition: So what?

VI. Conclusion & Future Work

i. QED: We built useful patches by considering a socio-technical information environment

ii. ZOOM OUT.

1. Heterogeneous patches aren't just a solution for Requirements Traceability Questions.

2. Heterogeneous/Socio-Technical can be more than Issue/Comment/People

2.1. Source Code

2.2. Semantic Similarity

iii. Could machine learning make better patches, or assign better weights?

iv. What would a useful everyday took look like, applying these patches?