Crawling Strategy: Algorithm for Web Crawling.

1. Create a list of all sites restricted in robots.txt for the given neu.edu and northeastern.edu domains
2. A) Start Loop until frontier queue (Breadth First Search) not empty OR the list of visited sites is 100.
3. Connect to given URL in frontier and find its Content-type.
4. If Content type is “pdf” or “text/html”
5. Parse links on the connected page into their canonical form
6. Put them in frontier if they are NOT visited and NOT in robots.txt and exclude links outside above said domains
7. Create a HASHMAP of main link and the outgoing links obtained in Step (2 C)
8. Write to a text file. End Loop
9. End Algorithm.

Strengths:

The following canonicalization are implemented in the crawler

1. Trailing “/” or “#” or “?” on every link was removed.

http://somepage.com/my\_page.html? == > http://somepage.com/my\_page.html

1. Removing duplicate slashes

http://somepage.com//my\_page.html == > http://somepage.com/my\_page.html

1. Converting the scheme and host to lower case

HTTP://somePaGe.com//my\_page.html == > http://somepage.com//my\_page.html

Weakness / Shortcomings:

1. Sorting the query parameters: both the below links are same

http://www.example.com/display?lang=en&article=fred → http://www.example.com/display?article=fred&lang=en

1. Removing the fragment.

http://www.example.com/bar.html#section1 → http://www.example.com/bar.html

1. Replacing IP with domain name.

http://208.77.188.166/ → http://www.example.com/

1. Limiting protocols.

https://www.example.com/ → http://www.example.com/

1. Removing or adding “www” as the first domain label.

http://www.example.com/ → http://example.com/

1. Removing default query parameters.

http://www.example.com/display?id=&sort=ascending → http://www.example.com/display

No, I won’t use this program in a production setting due to following reasons:

1. We have not implemented muti-threading concept in the program. Considering the size of the whole web, it would take forever to crawl even 1% of the web pages totally available.
2. Even if we implement muti-threading we need to distribute our crawler on several machines for different sites to improve performance and scale as the web grows.
3. The canonicalization is not completely implemented. So there could be sites which may contain repeated links in the output document. The weaknesses stated above add to the reasons why I won’t use this web crawler in a production setting.
4. We are storing the outgoing links from the site. It would had been better if we would have stored incoming links to the page. So, with the help of the data generated we might have more easily implemented the page rank algorithm.

For an industry grade web crawler should not have the shortcomings mentioned above. And they should implement the 3 points mentioned in the above production setting column.

References :

<http://en.wikipedia.org/wiki/URL_normalization>