(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2017년08월16일
(11) 등록번호 10-1768181
(24) 등록일자 2017년08월08일

(51) 국제특허분류(Int. Cl.)
G06F 17/30 (2006.01) *H04L 12/24* (2006.01)
H04L 29/08 (2006.01)
(52) CPC특허분류
G06F 17/30887 (2013.01)
H04L 41/0293 (2013.01)
(21) 출원번호 10-2017-7000222
(22) 출원일자(국제) 2014년06월26일
심사청구일자 2017년01월04일
(85) 번역문제출일자 2017년01월04일
(65) 공개번호 10-2017-0007521
(43) 공개일자 2017년01월18일
(86) 국제출원번호 PCT/CN2014/080801
(87) 국제공개번호 WO 2015/196405
국제공개일자 2015년12월30일
(56) 선행기술조사문헌
US07054917 B1

(73) 특허권자
구글 인코포레이티드
미국 캘리포니아 마운틴 뷰 엠피시어터 파크웨이 1600 (우:94043)
(72) 발명자
맹 비핑
중국 베이징 100084 하이뎬구 중관춘 이스트 로드 1 칭화 사이언스 파크 빌딩 6
이스템 마이클
미국 캘리포니아 마운틴 뷰 엠피시어터 파크웨이 1600
(뒷면에 계속)
(74) 대리인
박장원

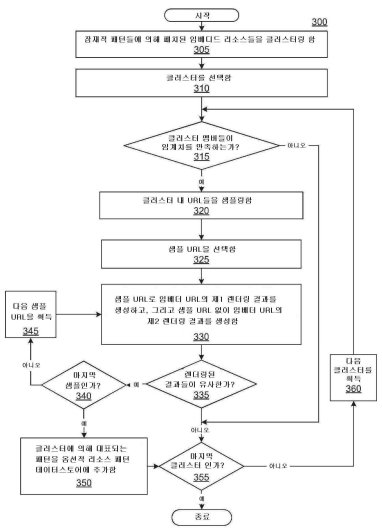
전체 청구항 수 : 총 20 항 심사관 : 이복현

(54) 발명의 명칭 **최적화된 브라우저 렌더링 프로세스**

(57) 요 약

구현들은 컨텐츠 중립적 임베디드 항목들을 식별함으로써 그리고 컨텐츠 중립적 항목들을 페칭하지 않고 웹 페이지를 렌더링함으로써 브라우저 렌더링 프로세스를 최적화한다. 예시적 방법은 페치 레코드들에 저장된 다수의 URL들에 공통된 URL 패턴을 식별하는 단계, 그리고 다수로부터 URL들의 샘플을 선택하는 단계를 포함한다. 방법은 또한 샘플 내 각각의 URL에 대하여 URL에 대한 컨텐츠를 사용하여 제1 렌더링 결과를 생성하고, URL에 대한 컨텐츠를 사용하지 않고 제2 렌더링 결과를 생성하며, 제1 렌더링 결과와 제2 렌더링 결과를 비교함으로써 URL에 대한 유사성 점수를 계산하여 유사성 점수가 유사성 임계치보다 클 때 URL이 옵션적임을 결정하는 것을 포함한다. 방법은 또한 샘플 내 URL들의 대다수가 옵션적일 때, 옵션적 리소스 패턴들의 데이터 스토어에 URL 패턴을 저장하는 것을 포함한다.

대 표 도 - 도3



(52) CPC특허분류
H04L 67/02 (2013.01)
(72) 발명자
슈 휘
미국 캘리포니아 마운틴 뷰 엠피시어터 파크웨이 1600
조우 샤오보
중국 베이징 100084 하이뎬구 중관춘 이스트 로드 1 칭화 사이언스 파크 빌딩 6

명 세 서

청구범위

청구항 1

컴퓨터 시스템으로서,

적어도 하나의 프로세서와; 그리고

명령어들을 저장한 메모리를 포함하고, 상기 명령어들은 상기 적어도 하나의 프로세서에 의해 실행될 때, 상기 시스템으로 하여금:

페치 레코드들에 저장된 복수의 URL들에 공통되는 URL 패턴을 식별하고;

상기 복수의 URL들로부터 URL의 샘플을 선택하고;

상기 샘플 내 각각의 URL에 대해:

상기 URL에 대한 컨텐츠를 사용하여 상기 URL의 임베더(embedder)에 대한 제1 렌더링 결과를 생성하고,

상기 URL에 대한 컨텐츠를 사용함이 없이 상기 임베더에 대한 제2 렌더링 결과를 생성하고,

상기 제1 렌더링 결과를 상기 제2 렌더링 결과와 비교함으로써 상기 URL에 대한 유사성 점수를 계산하고,

상기 유사성 점수가 유사성 임계치보다 클 때 상기 URL이 옵션적임을 결정함으로써, 상기 URL이 옵션적인지를 결정하고; 그리고

상기 샘플 내 상기 URL들의 미리 정해진 양(quantity)이 옵션적일 때, 옵션적 리소스 패턴들의 데이터 스토어에 상기 URL 패턴을 저장하는 것을 특징으로 하는 컴퓨터 시스템.

청구항 2

제1항에 있어서,

상기 명령어들은, 적어도 하나의 프로세서에 의해 실행될 때 상기 시스템으로 하여금:

요청된 임베디드 리소스의 컨텐츠에 대한 요청을 수신하고;

상기 요청된 임베디드 리소스가 상기 데이터 스토어 내 상기 패턴과 매치(match)하는지를 결정하고; 그리고

상기 요청된 임베디드 리소스가 상기 패턴과 매치할 때, 상기 요청된 임베디드 리소스가 옵션적이라는 표시를 반환하도록 하는 명령어들을 더 포함하는 것을 특징으로 하는 컴퓨터 시스템.

청구항 3

제2항에 있어서,

상기 표시는 URL 미발견 오류(URL not found error)인 것을 특징으로 하는 컴퓨터 시스템.

청구항 4

제1항에 있어서,

상기 URL들의 양은 상기 샘플 내 URL들의 양과 동일한 것을 특징으로 하는 컴퓨터 시스템.

청구항 5

제1항에 있어서,

상기 페치 레코드들에 저장된 상기 복수의 URL들에 공통되는 상기 URL 패턴을 식별하는 단계는:

상기 URL로부터 쿼리 문자열의 적어도 일부를 제거함으로써 상기 페치 레코드들에 저장된 각각의 URL들에 대한 그룹 URL을 생성하는 것과; 그리고

그룹 URL에 의해 상기 URL들을 클러스터링하는 것을 포함하는 것을 특징으로 하는 컴퓨터 시스템.

청구항 6

제1항에 있어서,

상기 페치 레코드들에 저장된 상기 복수의 URL들에 공통되는 상기 URL 패턴을 식별하는 단계는:

상기 URL로부터 쿼리 문자열의 적어도 일부를 제거함으로써 상기 페치 레코드들에 저장된 각각의 URL들에 대한 그룹 URL을 생성하는 것과;

그룹 URL에 의해 상기 URL들을 클러스터링하는 것과;

가장 많은 수의 멤버들을 가진 클러스터의 그룹 URL을 상기 URL 패턴으로 선택하는 것을 포함하는 것을 특징으로 하는 컴퓨터 시스템.

청구항 7

제1항에 있어서,

상기 유사성 점수를 계산하는 단계는:

상기 제1 렌더링 결과의 DOM 트리(DOM tree) 및 상기 제2 렌더링 결과의 DOM 트리에 대한 최장 공통 시퀀스(longest common sequence)를 결정하는 것과; 그리고

상기 유사성 점수를 결정하기 위해 상기 최장 공통 시퀀스를 사용하는 것을 포함하는 것을 특징으로 하는 컴퓨터 시스템.

청구항 8

방법으로서,

적어도 하나의 프로세서를 사용하여, 페치 레코드들에 저장된 복수의 URL들에 공통되는 URL 패턴을 식별하는 단계와;

적어도 하나의 프로세서를 사용하여, 상기 복수의 URL들로부터 URL들의 샘플을 선택하는 단계와;

상기 샘플의 각각의 URL에 대하여:

상기 URL에 대한 컨텐츠를 사용하여 상기 URL의 임베더에 대한 제1 렌더링 결과를 생성하고 그리고 상기 URL에 대한 컨텐츠를 사용함이 없이 상기 임베더에 대한 제2 렌더링 결과를 생성하고, 그리고

상기 제1 렌더링 결과와 상기 제2 렌더링 결과를 비교함으로써 상기 URL에 대한 유사성 점수를 계산 - 상기 URL은 상기 유사성 점수가 유사성 임계치보다 클 때 옵션적임 - 함으로써, 상기 URL이 옵션적인지를 결정하는 단계와; 그리고

상기 샘플 내 URL들의 대다수(majority)가 옵션적일 때, 옵션적 리소스 패턴들의 데이터 스토어에 상기 URL 패턴을 저장하는 단계를 포함하는 것을 특징으로 하는 방법.

청구항 9

제8항에 있어서,

요청된 임베디드 리소스의 컨텐츠에 대한 요청을 수신하는 단계와;

상기 요청된 임베디드 리소스가 상기 데이터 스토어 내 상기 패턴과 매치하는지를 결정하는 단계와; 그리고

상기 요청된 임베디드 리소스가 상기 패턴과 매치할 때, 상기 요청된 임베디드 리소스가 옵션적이라는 표시를 반환하는 단계를 더 포함하는 것을 특징으로 하는 방법.

청구항 10

제8항에 있어서,

상기 샘플 내 URL들이 옵션적인지를 결정하기 전에 상기 복수의 URL들에 의해 표시된 양이 사이즈 임계치를 초과함을 결정하는 단계를 더 포함하는 것을 특징으로 하는 방법.

청구항 11

제8항에 있어서,

상기 샘플 내 모든 URL들이 옵션적일 때, 옵션적 리소스 패턴들의 상기 데이터 스토어에 상기 URL 패턴을 저장하는 단계를 더 포함하는 것을 특징으로 하는 방법.

청구항 12

제8항에 있어서,

상기 페치 레코드들 내 상기 복수의 URL들에 공통되는 상기 URL 패턴을 식별하는 단계는:

상기 페치 레코드들 내 각각의 URL들에 대해, 상기 URL로부터 쿼리 문자열의 적어도 일부를 제거함으로써 상기 페치 레코드 내 상기 URL에 대한 그룹 URL을 생성하는 것과; 그리고

상기 그룹 URL에 의해 클러스터링을 하는 것을 포함하는 것을 특징으로 하는 방법.

청구항 13

제8항에 있어서,

상기 페치 레코드 내 상기 복수의 URL들에 공통되는 상기 URL 패턴을 식별하는 단계는:

상기 페치 레코드들 내 각각의 URL들에 대해, 상기 URL로부터 쿼리 문자열의 적어도 일부를 제거함으로써 상기 페치 레코드들 내 상기 URL에 대한 그룹 URL을 생성하는 것과;

상기 그룹 URL에 의해 클러스터링을 하는 것과; 그리고

클러스터 내 멤버들의 양이 임계치를 충족할 때, 상기 클러스터의 상기 그룹 URL을 선택하는 것을 포함하는 것을 특징으로 하는 방법.

청구항 14

제8항에 있어서,

상기 유사성 점수를 계산하는 단계는:

상기 제1 렌더링 결과의 DOM 트리 및 상기 제2 렌더링 결과의 DOM 트리에 대한 최장 공통 시퀀스를 결정하는 것과; 그리고

상기 유사성 점수를 결정하기 위해 최장 공통 시퀀스를 사용하는 것을 포함하는 것을 특징으로 하는 방법.

청구항 15

방법으로서:

웹페이지를 렌더링하기 위한 요청을 수신하는 단계와;

적어도 하나의 프로세서를 사용하여, 페치를 요하는 상기 웹페이지 내 적어도 하나의 임베디드 리소스를 식별하는 단계와;

상기 임베디드 리소스에 대한 URL이 옵션적 리소스들을 식별하는 데이터 스토어 내 패턴과 매치함을 결정함으로써, 상기 임베디드 리소스가 옵션적 리소스임을 결정하는 단계와; 그리고

상기 적어도 하나의 프로세서를 사용하여, 상기 임베디드 리소스에 대한 컨텐츠를 페칭함이 없이는 상기 임베디드 리소스가 이용 불가능한 것처럼 상기 웹페이지를 렌더링하는 단계를 포함하는 것을 특징으로 하는 방법.

청구항 16

제15항에 있어서,

서비스로부터의 옵션적 리소스들에 대한 패턴들의 데이터 스토어를 수신하는 단계를 더 포함하는 것을 특징으로 하는 방법.

청구항 17

제15항에 있어서,

상기 임베디드 리소스로 렌더링된 임베더 웹 페이지의 렌더링 결과가 상기 임베디드 리소스 없이 렌더링된 상기 임베더 웹 페이지의 렌더링 결과와 유사할 때, 상기 임베디드 리소스는 옵션적인 것을 특징으로 하는 방법.

청구항 18

제15항에 있어서,

상기 임베디드 리소스가 옵션적 리소스임을 결정하는 단계는:

상기 URL로부터의 쿼리 문자열을 제거함으로써 상기 임베디드 리소스에 대한 상기 URL을 재작성(rewriting) 하는 것과; 그리고

상기 재작성된 URL을 옵션적 리소스들의 상기 데이터 스토어 내 상기 패턴과 매치시키는 것을 포함하고, 상기 재작성된 URL이 상기 패턴과 매치할 때 상기 임베디드 리소스는 옵션적인 것을 특징으로 하는 방법.

청구항 19

제15항에 있어서,

상기 임베디드 리소스가 옵션적 리소스임을 결정하는 단계는:

상기 URL로부터 쿼리 문자열의 적어도 일부를 제거함으로써 상기 임베디드 리소스에 대한 상기 URL을 재작성 하는 것과; 그리고

상기 재작성된 URL을 옵션적 리소스들의 상기 데이터 스토어 내 상기 패턴과 매치시키는 것을 포함하고, 상기 재작성된 URL이 상기 패턴과 매치할 때 상기 임베디드 리소스는 옵션적인 것을 특징으로 하는 방법.

청구항 20

제15항에 있어서,

옵션적 리소스들의 상기 데이터 스토어는 렌더링 결과들의 비교에 기초하여 채워지는(populated) 것을 특징으로 하는 방법.

발명의 설명

배 경 기 술

[0001] 많은 웹 페이지들이 동적으로 생성된다. 즉, 이는 브라우저 사용자가 마지막으로 관람한 페이지가 브라우저 렌더링 프로세스로 인하여 적어도 부분적으로 생성된다는 의미이다. 스크립트, 이미지들, 스타일 시트들과 같은 임베디드 객체들은 종종 웹페이지에 내장되며, 렌더링된 웹페이지의 컨텐츠 및 모양(appearance)에 영향을 미치거나, 또는 동적 컨텐츠 생성과 관련없는 다른 목적들을 위해 포함될 수 있다. 예를 들어 Google Analytic JavaScript 코드는, 웹사이트로의 트래픽의 추적을 돕지만 컨텐츠에는 영향을 미치지 않는 임베디드 객체일 수 있다. 임베디드 객체들 자체들은 추가적인 임베디드 객체들을 포함할 수 있다. 마찬가지로, 어떤 임베디드 객체들은 숨겨져 있어 사용자들에게 보이지 않을 수 있다. 웹페이지는 수백 가지의 임베디드 객체들을 가질 수 있으며, 브라우저의 렌더링 엔진이 사용자를 위해 웹페이지를 렌더링 하는데 걸리는 시간의 대부분은 브라우저가 이들 임베디드 리소스들을 페칭(fetching)하는 것을 대기하는데 소비된다.

발명의 내용

[0002] 구현들은 옵션적(optional) 임베디드 리소스들을 식별하고 렌더링 엔진이 이러한 리소스들을 스킵하도록 하는바, 이는 렌더링 시간을 향상시킨다. 옵션적 리소스들은 렌더링된 웹페이지의 컨텐츠에 의미있거나

(meaningful) 또는 유의미한(significant) 방식으로 영향을 주지 않고 스킵 될 수 있는 임베디드 객체들을 나타낸다. 브라우저는 렌더링 프로세스 중에 이러한 옵션적 임베디드 리소스들을 스킵할 수 있으며, 결과적으로 로딩 시간이 빨라진다.

[0003] 한 가지 측면(aspect)에서, 시스템은 적어도 하나의 프로세서와, 상기 적어도 하나의 프로세서에 의해 실행될 때 시스템으로 하여금 동작들을 수행하게 하는 명령어들을 저장하는 메모리를 포함한다. 상기 동작들은 페치 레코드들 내에 저장된 복수의 URL들에게 공통된 URL패턴을 식별하는 단계와, 상기 복수의 URL들로부터 URL들의 샘플을 선택(select)하는 단계를 포함할 수 있다. 상기 동작들은 또한, 상기 샘플 내 각각의 URL에 대하여, 상기 URL에 대한 컨텐츠를 사용하여 상기 URL의 임베더에 대한 제1 렌더링 결과를 렌더링하는 단계, 상기 URL에 대한 컨텐츠를 사용하지 않고 상기 URL의 임베더에 대한 제2 렌더링 결과를 렌더링하며, 상기 제1 렌더링 결과를 상기 제2 렌더링 결과와 비교하여 상기 URL에 대한 유사성 점수를 계산하는 단계에 의해 상기 URL이 옵션적인 것인지 여부를 결정하는 것을 포함한다. 이 방법은 또한 상기 유사성 점수가 유사성 임계치보다 높을 때, 상기 URL이 옵션적인 것으로 결정하는 단계와, 상기 샘플 내 미리 정해진 수량의 URL들이 옵션적인 것일 때 옵션적 리소스 패턴들의 데이터 스토어에 상기 URL 패턴을 저장하는 단계를 포함한다.

[0004] 본 명세서에 설명된 발명의 구현들 중 하나 이상은 다음 특징 중 하나 이상을 포함할 수 있다. 예를 들어 상기 동작들은 또한, 요청된 임베디드 리소스의 컨텐츠에 대한 요청을 수신하는 단계, 상기 요청된 임베디드 리소스가 데이터 스토어 내 패턴과 매치하는지(match) 여부를 결정하는 단계, 그리고 상기 요청된 임베디드 리소스가 상기 패턴과 매치할 때 상기 요청된 임베디드 리소스가 옵션적인 것이라는 표시를 반환하는 단계를 포함할 수 있다. 일부 구현들에서, 상기 표시는 URL을 비발견 오류일 수 있다. 또 다른 예로서, 페치 레코드들에 저장된 상기 복수의 URL들에 공통된 상기 URL 패턴을 식별하는 단계는 상기 URL로부터의 쿼리 문자열의 적어도 일부를 제거함으로써 페치 레코드들에 저장된 각각의 URL들에 대한 그룹 URL을 생성하는 단계, 그리고 그룹 URL 별로 상기 URL들을 클러스터링 하는 단계를 포함할 수 있다. 또 다른 예로서, 페치 레코드들에 저장된 상기 복수의 URL들에 공통된 상기 URL 패턴을 식별하는 단계는 상기 URL로부터의 쿼리 문자열의 적어도 일부를 제거함으로써 페치 레코드들에 저장된 각각의 URL에 대한 그룹 URL을 생성하는 단계, 그룹 URL별로 상기 URL들을 클러스터링 하는 단계, 그리고 클러스터의 가장 큰 숫자의 구성원들 수를 가진 그룹URL을 상기 URL 패턴으로 선택하는 단계를 포함할 수 있다. 또 다른 예로서, 상기 유사성 점수를 계산하는 단계는 상기 제1 렌더링 결과의 DOM 트리와 상기 제2 렌더링 결과의 DOM 트리에 대한 최장 공통 시퀀스(longest common sequence)를 결정하는 단계, 그리고 상기 유사성 점수를 결정하기 위해 상기 최장 공통 시퀀스를 사용하는 단계를 포함할 수 있다.

[0005] 또 다른 측면에서, 컴퓨터 구현 방법은 페치 레코드들에 저장된 복수의 URL들에 공통된 URL 패턴을 식별하는 단계와, 상기 적어도 하나의 프로세서를 사용하여 상기 복수의 URL들로부터 URL들의 샘플을 선택하는 단계를 포함할 수 있다. 이 방법은 또한, 상기 URL들의 샘플 내 각각의 URL에 대하여, 상기 URL에 대한 컨텐츠를 사용하여 URL의 임베더에 대한 제1 렌더링 결과를 생성하는 단계, 상기 URL에 대한 컨텐츠를 사용하지 않고 상기 임베더에 대한 제2 렌더링 결과를 생성하는 단계, 그리고 상기 제1 렌더링 결과를 상기 제2 렌더링 결과와 비교함으로써 유사성 점수를 계산하는 단계들에 의해, 상기 URL이 옵션적인 것인지 여부를 결정하는 단계를 포함할 수 있으며, 상기 URL은 유사성 점수가 유사성 임계치보다 클 때 옵션적인 것이다. 이 방법은 또한, 상기 샘플 내 대부분의 URL들이 옵션적인 것일 때 옵션적 리소스 패턴들의 데이터 스토어에 상기 URL 패턴을 저장하는 단계를 포함할 수 있다.

[0006] 본 명세서에 설명된 발명의 구현들 중 하나 이상은 다음 특징들 중 하나 이상을 포함할 수 있다. 예를 들어 이 방법은 또한, 상기 샘플 내 URL들이 옵션적인 것인지 여부를 결정하는 단계 이전에, 상기 복수의 URL들에 의해 표현된 수량이 사이즈 임계치를 초과하는 것을 결정하는 단계를 포함할 수 있다. 또 다른 예로서, 이 방법은 상기 샘플 내 모든 URL들이 옵션적인 것일 때, 상기 URL 패턴을 옵션적 리소스 패턴들의 데이터 스토어에 저장하는 단계를 포함할 수 있다. 또 다른 예로서, 상기 페치 레코드들 내 복수의 URL들에 공통된 URL 패턴을 식별하는 단계는, 상기 페치 레코드들 내 각각의 URL들에 대하여 상기 URL로부터의 쿼리 문자열의 적어도 일부를 제거함으로써 상기 페치 레코드 내 URL에 대한 그룹 URL을 생성하는 단계, 그룹 URL 별로 클러스터링하는 단계, 그리고 클러스터 내 구성원들의 수량이 임계치를 충족할 때 상기 클러스터의 그룹 URL을 선택하는 단계를 포함할 수 있다.

[0007] 또 다른 측면에서, 방법은 웹페이지를 렌더링하라는 요청을 수신하는 단계, 페칭을 요구하는 웹 페이지 내 적어도 하나의 임베디드 리소스를 식별하는 단계, 임베디드 리소스가 옵션적 리소스인 것을 결정하는 단계, 그리고 마치 상기 임베디드 리소스에 대한 컨텐츠를 페칭하지 않고는 상기 임베디드 리소스는 이용 가능하지 않은 것처

럼 상기 웹 페이지를 렌더링 하는 단계를 포함할 수 있다.

[0008] 본 명세서에 설명된 발명의 구현들 중 하나 이상은 다음 특징들 중 하나 이상을 포함할 수 있다. 예를 들어, 상기 임베디드 리소스가 옵션적인 것임을 결정하는 단계는, 상기 임베디드 리소스에 대한 URL이 옵션적 리소스들의 데이터 스토어 내 패턴과 매치함을 결정하는 단계를 포함할 수 있다. 또 다른 예로서, 상기 방법은 또한, 서비스로부터 옵션적 리소스들에 대한 패턴들의 데이터 스토어를 수신하는 단계를 포함할 수 있으며, 상기 임베디드 리소스가 옵션적 리소스인 것을 결정하는 단계는 상기 임베디드 리소스의 URL이 상기 데이터 스토어 내 패턴과 매치한다는 것을 결정하는 단계를 포함한다. 또 다른 예로서, 상기 임베디드 리소스가 옵션적 리소스인 것을 결정하는 단계는, 상기 임베디드 리소스에 대한 URL로부터 쿼리 문자열 부분을 제거함으로써 상기 URL을 재기록(rewrite)하는 단계와, 상기 재기록된 URL을 옵션적 리소스들의 데이터 스토어 내 패턴과 매치하는 단계를 포함하며, 상기 다시 작성된 URL이 상기 패턴과 매칭될 때 상기 임베디드 리소스는 옵션적인 것이다. 또 다른 예로, 상기 임베디드 리소스가 옵션적 리소스인 것으로 결정하는 단계는, 상기 임베디드 리소스에 대한 URL을 상기 URL로부터의 쿼리 문자열의 적어도 일부를 제거함으로써 다시 작성하는 단계와, 상기 다시 작성된 URL을 옵션적 리소스들의 데이터 스토어 내 패턴과 매치하는 단계를 포함할 수 있으며, 상기 다시 작성된 URL이 상기 패턴과 매치할 때 상기 임베디드 리소스는 옵션적인 것이다.

[0009] 또 다른 측면에서, 비-일시적인 컴퓨터-판독가능 매체는 컴퓨터 시스템이 전술된 방법들 중 하나 이상을 수행하게 하는 기판 내에 형성된, 적어도 하나의 프로세서에 의해 실행가능 한 명령어들을 포함할 수 있다.

[0010] 본 명세서에 설명된 발명의 하나 이상의 구현들은 다음 이점들 중 하나 이상을 실현하도록 구현될 수 있다. 일 예로서, 옵션적 임베디드 리소스들은 스킵하는 것은 상기 렌더링 엔진이 컨텐츠의 손실 없이 더 빠르게 상기 웹 페이지를 렌더링할 수 있도록 허용한다. 브라우저는 옵션적 임베디드 리소스를 페칭하는 것을 스킵할 수 있을 뿐 아니라, 상기 브라우저는 상기 스킵된 임베디드 리소스에 내장된 객체들을 페칭하는 것도 스킵한다. 개시된 구현들은 이와 같이, 네트워크 대역폭 및 배터리 소모를 절약한다. 웹페이지들은 수 백개의 임베디드 객체들을 포함할 수 있으므로, 이러한 최적화들은 실속이 있다

도면의 간단한 설명

[0011] 도 1은 개시된 발명에 따른 시스템 예를 나타낸다.

도 2는 구현에 따른, 렌더링 된 웹페이지에 대한 렌더링 결과의 블록 다이어그램이다.

도 3은 구현에 따른, 옵션적 리소스 패턴들을 식별하는 프로세스 예를 나타낸 흐름도이다.

도 4는 구현에 따른, 2개의 렌더링 결과들의 유사 여부를 결정하는 프로세스 예를 나타낸 흐름도이다.

도 5는 구현에 따른, 최적화된 렌더링을 위한 정보를 제공하기 위한 프로세스 예를 나타낸 흐름도이다.

도 6은 본 명세서에 설명된 기법들을 구현하는데에 사용되는 컴퓨터 디바이스의 예를 나타낸다.

도 7은 본 명세서에 설명된 기법들을 구현하는데에 사용되는 분산형 컴퓨터 디바이스의 예를 나타낸다.

발명을 실시하기 위한 구체적인 내용

[0012] 개시된 구현들은 옵션적 임베디드 객체들, 또는 렌더링된 웹페이지의 컨텐츠에 유의미하게 영향을 미치지 않는 임베디드 객체들을 식별하며, 브라우저 렌더링 엔진이 옵션적 임베디드 객체들을 페칭하지 않고 이런 웹 페이지들을 렌더링할 수 있도록 한다. 이는 렌더링 시간과 그리고 페칭 시스템의 로드를 개선하는바, 이는 스마트폰들과 타블렛들과 같은 제한된 리소스들을 가진 컴퓨팅 디바이스들에게 중요하다. 임베디드 리소스는 옵션적 임베디드 객체로 렌더링된 임베더 웹페이지의 렌더링 결과와 옵션적 임베디드 객체 없이 렌더링된 임베더 웹페이지의 렌더링 결과가 유사할 때 최적화된 렌더링의 후보가 될 수 있다. 유사성은 페이지 내 토큰들의 유사성, 아웃링크들의 유사성, 페이지 레이아웃의 유사성 등으로 결정될 수 있다. 일부 구현들에서. 시스템은 상기 렌더링 결과들의 유사 여부를 판단하기 위해 최장 공통 시퀀스 측정을 사용할 수 있다. 일부 구현들에서, 스냅 샷들의 주된 컴포넌트들이 유사성 임계치를 만족하는 유사성 점수를 가질 때, 상기 스냅 샷들은 유사한 것으로 여겨진다. 시스템이 임베디드 리소스가 옵션적인 것이라고 결정할 경우, 상기 시스템은 상기 임베디드 리소스(예를 들어 이의 URL)를 옵션적 임베디드 리소스들의 리스트에 저장할 수 있다. 일부 구현들에서, 시스템은 다른 옵션적 임베디드 리소스들을 식별하기 위해 사용될 수 있는 패턴을 식별하기 위해 옵션적 리소스 URL들을 종합할 수 있다.

[0013] 브라우저가 웹페이지 렌더링을 시작할 때, 임베디드 리소스들 중 어느것이 옵션적인지를 서비스에 물어볼(ask) 수 있다. 일부 구현들에서, 이는 페치 요청의 일부로서 행해질 수 있다. 상기 서비스는 상기 요청된 임베디드 리소스들의 URL을 정확히 매치하거나 또는 데이터 스토어의 패턴과 매치함으로써, 옵션적 임베디드 객체들의 데이터 스토어에 매치할 수 있다. 상기 서비스가 매치를 발견할 경우, 상기 서비스는 어떤 임베디드 리소스들이 옵션적인 것인지 브라우저에게 알려줄 수 있으며, 상기 브라우저는 이러한 임베디드 리소스들을 페칭하지 않고 웹페이지를 렌더링할 수 있다. 이러한 최적화된 브라우저 렌더링은 상기 브라우저 렌더링 프로세스를 더욱 리소스 효율적(resource efficient)으로 만든다. 일부 구현들에서, 옵션적 임베디드 리소스 데이터 스토어는 브라우저 유형들(예를 들어 모바일 브라우저들 및 데스크 탑 브라우저들) 간을 구분할 수 있다.

[0014] 도 1은 예시적 구현에 따른 시스템의 블록 다이어그램이다. 시스템(100)은 옵션적 임베디드 리소스들과 옵션적 리소스 패턴들을 식별하기 위해 사용될 수 있고, 브라우저 렌더링 프로세스를 최적화하기 위해 상기 패턴들을 사용할 수 있다. 시스템(100)은 다수의 다양한 디바이스들의 형태들을 취하는 컴퓨팅 디바이스들을 포함할 수 있다. 예를 들어, 시스템(100)은 도 6에 도시된 바와 같은 컴퓨터 디바이스(600)와 도 7에 도시된 바와 같은 컴퓨터 디바이스(700)의 예시들인 컴퓨팅 디바이스들을 포함할 수 있다.

[0015] 시스템(100)은 배치 렌더링 시스템(110)을 포함할 수 있다. 배치 렌더링 시스템(110)은 표준서버, 그러한 서버들의 그룹, 클라이언트-서버 시스템, 또는 렉 서버 시스템이 될 수 있다. 또한, 배치 렌더링 시스템(110)은 개인용 컴퓨터에서도 구현될 수 있다. 배치 렌더링 시스템(110)은 메모리에 저장된 하나 이상의 기계 실행가능 명령어들 또는 소프트웨어의 조각들, 펌웨어, 또는 이들의 조합을 실행하도록 구성된 하나 이상의 프로세서들을 포함할 수 있다. 예를 들어, 배치 렌더링 시스템(110)은 렌더링 엔진들(120), 페치 서비스(122), 웹-크롤링 엔진(124), 그리고 옵션적 리소스 식별 엔진(126)을 포함할 수 있다. 배치 렌더링 시스템(110)은 예를 들어, 인덱싱 프로세스의 일부로서 렌더링된 웹페이지를 배치모드로 생산할 수 있고, 이 절차에서 상기 배치 렌더링 시스템(110)은 옵션적 리소스 패턴들(130)을 생성하기 위해 상기 배치 렌더링 시스템(110)이 사용할 수 있는 페치 레코드들(132)을 생산할 수 있다.

[0016] 다운 스트림 프로세스는 상기 배치 렌더링 시스템(110)이 배치모드로 웹 페이지를 렌더링하도록 요청할 수 있다. 일부 구현들에서, 상기 배치 렌더링 시스템(110)은 옵션적 리소스 식별 엔진(126) 또는 인덱싱 엔진(도시되지 않음)과 같은 다운 스트림 프로세스들을 포함할 수 있다. 일부 구현들에서, 다운 스트림 프로세스는 배치 렌더링 시스템(110)과 상이한 컴퓨팅 디바이스 상에서 실행될 수 있다. 예를 들어, 다운 스트림 프로세스는 원격 절차 호출을 통해 배치 렌더링 시스템에 요청들을 송신하는 인덱싱 엔진 또는 광고 플랫폼일 수 있다. 다운 스트림 프로세스는 배치 렌더링 엔진들(120) 중 하나가 특정 웹페이지의 렌더링 결과를 생성하도록 요청할 수 있다. 각 배치 렌더링 엔진(120)은 배치 렌더링에 대한 최적화들과 함께, 개인 웹 브라우저에 대한 렌더러를 에뮬레이트 하도록 구성될 수 있다. 배치 렌더링 시스템(110)은 수천 개의 배치 렌더링 엔진들(120)을 포함할 수 있으며, 특정 요청에 응답하기 위해 배치 렌더링 엔진들(120) 중 하나를 선택하도록 로드-밸런싱(load-balancing)을 사용할 수 있다. 상기 요청된 웹 페이지는 스타일 시트들, 자바스크립트, 이미지들 등과 같은 임베디드 객체들을 포함할 수 있다. 배치 렌더링 엔진들(120)은 페치 서비스(122)를 이용하여 임베디드 객체들에 대한 컨텐츠를 요청할 수 있다.

[0017] 페치 서비스(122)는 어떤 임베디드 리소스들이 호스트 서버(190)로부터 페치 될 필요가 있는지, 어떤 임베디드 리소스들이 캐시로부터 반환될 수 있는지, 그리고 어떤 리소스들이 반환될 필요가 없는지 결정할 수 있다. 호스트 서버들(190)은 하나 이상의 웹페이지들 또는 하나 이상의 웹페이지들에 내장된 리소스들을 호스트하는 인터넷을 통해 액세스 가능한 임의의 유형의 컴퓨팅 디바이스일 수 있다. 임베디드 리소스가 페치될 필요가 있는 경우, 페치 서비스(122)는 종래의 기법들을 사용하여 웹-크롤링 엔진(124)을 통해 임베디드 객체에 대한 컨텐츠를 요청할 수 있다. 인덱싱 엔진과 같은 다운 스트림 프로세스는 또한, 웹-크롤링 엔진(124)를 통해 서버들(190)로부터 컨텐츠를 요청할 수 있다. 웹-크롤링 엔진(124)를 통한 페치 요청들의 결과로, 배치 렌더링 시스템(110)은 페치 레코드들(132)을 생성할 수 있다. 페치 레코드들(132)는 어떤 웹페이지들 및 임베디드 객체들이 호스트 서버들(190)로부터 요청되고 검색되는지에 대한 정보를 포함할 수 있다. 페치 레코드들(132)은 또한, 요청 시간, 임베더 웹페이지 등 추가적인 정보가 존재할 경우 이들을 포함할 수 있다.

[0018] 웹-크롤링 엔진(124), 배치 렌더링 엔진들(120), 그리고 페치 서비스(122)는 월드 와이드 웹(World Wide Web) 상에서 발견될 수 있는 웹 페이지들과 같은, 다수의 웹페이지들을 효과적으로 렌더링하기 위해 함께 동작한다. 웹페이지의 렌더링은 다운 스트림 요청 프로세스에 유용한(그렇지 않으면 이용가능 하지 않은) 다양한 데이터 요소들을 포함하는 렌더링 결과이다. 옵션적 리소스 식별 엔진(126)은 옵션적 리소스 패턴들(130)을 생성하기 위해 배치 렌더링 엔진들(120)을 사용할 수 있다. 옵션적 리소스 식별 엔진(126)은 페치 레코드들(132)을 입력

으로 사용하여, 주기적으로(예를 들어, 매일, 주 2회 등) 실행할 수 있다. 옵션적 리소스 식별 엔진(126)은 페치 레코드(132)를 분석할 수 있고, 이전 기간에 페치된 URL들에 대한 패턴들을 생성할 수 있다. 예를 들어, 옵션적 리소스 식별 엔진(126)은 페치 레코드들 내 URL들의 쿼리 문자열(query string)을 제거하여, 각각의 URL에 대한 그룹 URL을 생성할 수 있다. 쿼리 문자열은 상기 URL에서 물음표(?) 뒤에 오는 임의의 문자들일 수 있다. 일부 구현들에서, 상기 URL에 대한 그룹 URL을 생성하기 위해 쿼리 문자열의 일부만 제거될 수 있다. 그 다음, 옵션적 리소스 식별 엔진(126)은 상기 그룹 URL들을 정렬 또는 클러스터링할 수 있고, 어떤 그룹 URL들이 시간 기간 내에 가장 많은 페치 요청들의 수를 가졌는지를 결정할 수 있다.

[0019] 옵션적 리소스 식별 엔진(126)은 가장 많은 수의 페치 요청들을 갖는 그룹URL들을 잠재적 패턴들로서 선택할 수 있다. 이러한 패턴들은 가장 빈번하게 페치되었으므로, 브라우징 프로세스의 최적화를 위해 최상의 잠재력을 갖는 임베디드 리소스들을 나타낸다. 옵션적 리소스 식별 엔진은 상기 잠재적인 패턴과 매치하는 URL들 또는 잠재적인 패턴과 매치하는 URL들의 샘플이 옵션적인 것인지 여부를 결정할 수 있다. 예를 들어, 옵션적 리소스 식별 엔진(126)은 페치 레코드들(132)로부터의 잠재적 패턴과 매치하는 임베디드 리소스를 식별할 수 있고, 배치 렌더링 엔진들(120)이 임베디드 리소스에 대한 임베더 웹 페이지의 제1 렌더링 결과를 렌더링하도록 요청할 수 있다. 임베더 웹 페이지는 예컨대, 임베디드 리소스에 대한 페치 레코드로부터 식별될 수 있다. 그 다음, 옵션적 리소스 식별 엔진(126)은 상기 배치 렌더링 엔진들(120)이 상기 임베디드 리소스를 페치하지 않고 상기 임베더 웹페이지의 제2 렌더링 결과를 렌더링하도록 요청할 수 있다.

[0020] 임베더 웹페이지의 렌더링된 컨텐츠가 특정 임베디드 리소스를 스킵 함으로써 영향을 받았는지 판단하기 위해, 옵션적 리소스 식별 엔진(126)은 제1 렌더링 결과와 제2 렌더링 결과를 비교할 수 있다. 일부 구현들에서, 만약 컨텐츠가 유의미하게 영향을 받을 경우, 특정 임베디드 리소스는 요구되는 리소스들의 리스트에 추가될 수 있다. 만약 컨텐츠가 유의미하게 영향을 받지 않을 경우(예를 들어, 컨텐츠가 유사할 경우), 옵션적 리소스 식별엔진(126)은 임베디드 리소스를 옵션적 리소스로서 식별할 수 있다. 일부 구현들에서, 임베디드 리소스(예를 들어, 그것의 URL)는 옵션적 리소스 패턴들(130)과 같은 데이터 스토어에 저장될 수 있다. 일부 구현들에서, 임베디드 리소스는 옵션적 리소스들에 대한 패턴들을 결정하기 위해 나중에 사용되는, 옵션적 리소스들의 일시적 리스트에 저장될 수 있다. 옵션적 리소스 식별 엔진(126)은 페치 레코드들 내 잠재적인 패턴과 매치하는 모든 임베디드 리소스에 대하여, 또는 잠재적인 패턴과 매치하는 임베디드 리소스들의 샘플에 대하여 이 테스트(렌더링 결과들의 비교)를 수행할 수 있다. 일부 구현들에서, 잠재적인 패턴들과 매치하는 임의의 임베디드 리소스들이 관련된 경우, 상기 잠재적인 패턴은 옵션적 리소스 패턴이 아니다. 일부 구현들에서, 잠재적인 패턴과 매치하는 임베디드 리소스들의 작은 비율(예를 들어, 1% 이하)만이 요구되는 리소스들일 경우, 잠재적인 패턴은 옵션적 리소스 패턴이라 간주되며, 옵션적 리소스 식별 엔진(126)은 옵션적 리소스 패턴들(130) 내의 패턴으로써 잠재적인 패턴을 포함할 수 있다. 물론, 잠재적인 패턴과 매치하는 테스트된 임베디드 리소스들 모두가 옵션적인 것일 경우, 옵션적 리소스 식별 엔진(126)은 상기 잠재적인 패턴을 옵션적 리소스 패턴들(130)에 추가할 수 있다.

[0021] 일부 구현들에서, 옵션적 리소스 식별 엔진(126)은 또한, 옵션적 리소스 패턴(130) 내 패턴들이 여전히 옵션적인 것임을 검증할 수 있다. 예를 들어, 옵션적 리소스 식별 엔진(126)은 상기 패턴에 매치하는 URL들의 샘플을 선택하고, 렌더링 엔진이 상기 패턴과 매치하는 임베디드 리소스로 렌더링 결과를 렌더링하도록 요청하고, 그리고 상기 임베디드 리소스 없이 렌더링 결과를 렌더링하도록 요청하고, 상기 두 개의 렌더링 결과들을 비교할 수 있다. 옵션적 리소스 패턴들(130) 내 패턴과 매치하는 URL이 더 이상 옵션적인 것이 아닐 경우, 상기 패턴은 제거될 수 있다.

[0022] 일부 구현들에서, 옵션적 리소스 식별 엔진(126)은 풀 브라우저들과는 별도로 모바일 브라우저들에 대해 패턴이 옵션적인 것인지 여부를 결정할 수 있다. 예를 들어, 어떤 웹사이트들은 웹페이지의 모바일 버전과 풀 버전에 대해 다른 컨텐츠를 로딩하고, 임베디드 리소스는 모바일 브라우저에 대해서는 옵션적이지만 풀 브라우저에 대해서는 요구될 수 있다. 따라서, 옵션적 리소스 식별 엔진(126)은 임베디드 리소스에 대하여 네 개의 렌더링 결과들을 렌더링할 수 있다. 처음 두 개의 렌더링 결과들은 풀 브라우저를 에뮬레이트하는 렌더링 엔진에 의해 렌더링 될 수 있고, 마지막 두 개의 렌더링 결과들은 스마트폰 또는 타블렛과 같은 모바일 디바이스 상에서 모바일 브라우저를 에뮬레이트하는 렌더링 엔진에 의해 렌더링 될 수 있다. 따라서, 옵션적 리소스 패턴들(130)은 예를 들어, 모바일 브라우저에 대한 또는 풀 브라우저에 대한과 같은 브라우저 유형에 의해 패턴이 옵션적인 것인지 여부를 표시하는 데이터를 포함할 수 있다.

[0023] 간결성을 위해 도 1에는 나타나지 않았으나, 일부 구현들에서 배치 렌더링 시스템(110)은 두 가지 이상의 별개의 컴퓨팅 디바이스들에 분산될 수 있다. 또한, 배치 렌더링 엔진들(120), 페치 서비스(122), 웹-크롤링 엔진

(124) 및 옵션적 리소스 식별 엔진(126) 중 하나 이상은, 하나 이상의 컴퓨팅 디바이스들에 걸쳐 분산될 수 있다. 일부 구현들에서, 배치 렌더링 엔진들(120), 페치 서비스(122), 웹-크롤링 엔진(124) 및 옵션적 리소스 식별 엔진(126) 중 하나 이상은 메모리 또는 하드웨어 프로세서들과 같은 리소스들을 배치 렌더링 시스템(110)의 다른 컴포넌트들과 공유할 수 있다. 유사하게, 페치 레코드들(132) 및 옵션적 리소스 패턴들(130)은 또한, 다수의 컴퓨팅 디바이스들에 걸쳐 분산된 메모리에 저장될 수 있다. 일부 구현들에서, 배치 렌더링 시스템(110)의 다양한 컴포넌트들은 컴퓨팅 디바이스의 하드웨어 컴포넌트들을 공유할 수 있거나 또는 동일한 컴퓨팅 디바이스의 논리적 분할들일 수 있다.

[0024] 배치 렌더링 시스템(110)은 네트워크(160)를 통해 클라이언트(들)(180) 및 서버들(190)과 통신할 수 있다. 네트워크(160)는 예를 들어 인터넷 일 수 있거나 또는 네트워크(160)는 예를 들어 게이트웨이 디바이스들, 브릿지들, 스위치들, 및/또는 기타 등등을 사용하여 구현되는 유선 또는 무선 근거리 통신망(LAN), 광역 통신망(WAN), 이들의 조합 등이 될 수 있다. 네트워크(160)을 통해 배치 렌더링 시스템(110)은 클라이언트들(180) 및/또는 호스트 서버들(190)과 통신하고 이들에게/로부터 데이터를 전송할 수 있다. 예를 들어, 배치 렌더링 시스템(110)은 클라이언트(180)에게 옵션적 리소스 패턴들(130)을 제공할 수 있거나 또는 클라이언트(180)로부터 특정 임베디드 리소스가 옵션적인 것인지 여부를 묻는 요청을 수신할 수 있고, 옵션적 리소스 패턴들(130)에 근거하여 응답을 제공할 수 있다.

[0025] 클라이언트(180)는 개인용 웹 브라우저(140)가 설치된 도 6에 도시된 컴퓨팅 디바이스(600)와 같은 개인용 컴퓨팅 디바이스일 수 있다. 개인용 웹 브라우저들의 예시들은 스마트폰이나 타블렛과 같은 모바일 디바이스 또는 랩탑이나 데스크 탑과 같은 개인용 컴퓨터에 관계없이, Chrome, Safari, Internet Explorer, Firefox 등을 포함한다. 브라우저(140)는 웹 페이지를 렌더링할 때 특정 임베디드 리소스를 페치할 지 여부를 결정하기 위해 서비스를 사용하도록 구성될 수 있다. 일부 구현들에서, 시스템은 브라우저(140)에 의한 사용을 위해 클라이언트(180)에게 옵션적 리소스 패턴들(130)의 복사본을 푸쉬(push)할 수 있다. 이러한 의미에서, 서비스는 로컬 서비스일 수 있다. 일부 구현들에서, 브라우저(140)는 특정 임베디드 리소스가 옵션적 리소스 패턴들(130) 내 패턴들 중 하나와 매치하는지를 결정하기 위해, 서버-기반 서비스에 물어보도록 구성될 수 있다. 브라우저(140)는 웹 페이지를 렌더링할 때, 옵션적 임베디드 리소스들을 스킵하기 위해 서비스를 사용함으로써 최적화된 렌더링을 수행한다. 따라서, 브라우저(140)는 렌더링 시간을 향상하고, 컨텐츠가 영향을 받지 않는 리소스들을 보존하기 위해 임베디드 리소스들을 선택적으로 스킵하도록 구성될 수 있다.

[0026] 도 2는 구현에 따른, 배치 렌더링 엔진(120)에 의해 렌더링된 렌더링 결과(200)의 블록 다이어그램이다. 렌더링 결과(200)는 도 2에는 모두 도시되지 않았으나, 다양한 컴포넌트들을 포함할 수 있다. 예를 들어, 렌더링 결과(200)는 렌더링된 페이지의 이미지(205)를 포함할 수 있다. 이미지(205)는 웹 브라우저를 통해 사용자에게 디스플레이 디바이스를 통해 디스플레이될 수 있는 이미지일 수 있다. 이미지(205)는 예를 들어, 사용자에게 렌더링된 페이지의 썸네일을 디스플레이하기 위해, 그리고 어디서(예를 들어 이들의 x 및 y 좌표들에 기초하여) 웹페이지의 디스플레이 요소들이 발생하는지 결정하기 위해 사용될 수 있다. 렌더링 결과(200)는 또한, 문서객체모델(DOM) 트리(210)를 포함할 수 있다. DOM 트리(210)는 웹페이지의 HTML 구조를 나타낸다. 렌더링 결과(200)는 또한 레이아웃(215)을 포함한다. 레이아웃(215)은 일반적으로 웹 페이지의 요소들에 대한 박스를 포함하고, 상기 박스는 이미지(205) 내 요소의 x 및 y 좌표들을 명시한다. 따라서, 레이아웃(215)은 요소가 웹페이지 상 어디에서 발생하는지, 웹 페이지 상에서 요소가 얼마나 많은 공간을 차지하는지 등에 대한 표시를 제공한다. 따라서, 레이아웃(215)은 웹 페이지의 얼마나 많은 부분이 광고인지, 단락이 얼마나 눈에 잘 띄는지(예를 들어 above-the-fold 또는 below-the-fold), 요소가 가시적인지의 여부, 등등에 대한 정보를 제공한다. 즉, 레이아웃(215)은 렌더링된 웹페이지의 요소들에 대한 기하학적 정보를 제공한다. 렌더링 결과(200)는 또한, 에러들(220)을 포함할 수 있다. 에러들(220)은 자바스크립트와 같은 스크립트 객체들을 실행한 결과로서 직면하게 되는 에러들을 포함한다. 렌더링 결과(200)는 또한, 렌더링 중에 페치된 임베디드 리소스들(225)의 리스트를 포함하고, 렌더링 프로세스의 일부로서 생성된 다른 요소들(도시되지 않음)을 포함할 수 있다. 따라서, 렌더링 결과(200)는 단지 호스팅 서버로부터의 컨텐츠의 페치를 통해서는 이용가능하지 않은 정보를 제공한다. 옵션적 리소스 식별 엔진과 같은 다운 스트림 요청 프로세스들은, 예를 들어 스크립트 객체들을 실행하는 것이 렌더링된 웹페이지의 컨텐츠를 유의미하게 변화시키는지를 판단하기 위함과 같이 다양한 목적들을 위해 렌더링 결과 정보를 사용할 수 있다. 예를 들어, 웹 페이지의 광고부분의 컨텐츠는 중요하게 여겨지지 않을 수 있으며, 그래서 광고부분에서 차이가 있어도 URL은 여전히 옵션적인 것일 수 있다. 일부 구현들에서 오직 렌더링 결과의 주된 컴포넌트들 간 차이들만이 중요하다 여겨진다. 주된 컴포넌트들은 레이아웃(215) 내 가장 큰 박스들(예를 들어 가장 큰 높이와 넓이를 가진 박스들)일 수 있다.

[0027] 도 3은 구현에 따른, 옵션적 리소스 패턴들을 식별하기 위한 예시적 프로세스(300)를 나타내는 흐름도이다. 프로세스(300)는 도 1의 시스템(110)과 같은 시스템에 의해 수행될 수 있다. 시스템은 페치 레코드들로부터의 옵션적 리소스들에 대한 패턴들을 식별하기 위해 프로세스(300)를 사용할 수 있다. 시스템은 리소스를 사용하여 그리고 사용하지 않고 렌더링된, 임베더 웹페이지의 렌더링 결과 비교에 기초하여 옵션적 리소스들에 대한 패턴들을 결정할 수 있다. 상기 패턴들은 데이터 스토어에 저장될 수 있으며 풀 브라우저, 모바일 브라우저, 또는 배치 렌더링 엔진인지에 관계없이 브라우저가 옵션적 리소스들을 스킵함(예를 들어, 페치하지 않음)으로써 렌더링을 최적화할 수 있도록 하는 서비스의 일부로서 사용될 수 있다.

[0028] 프로세스(300)는 시스템이 페치된 임베디드 리소스들을 잠재적인 패턴들에 의해 클러스터링하면서 시작할 수 있다(305). 시스템은 프로세스(300)가 마지막으로 실행된 이래로 예를 들어, 하루, 이틀, 일주일과 같은 어떤 이전의 시간 기간에 페치된 임베디드 리소스들을 모두 선택할 수 있다. 상기 잠재적인 패턴들은 페치 레코드들에서 발견된 URL로부터 쿼리 문자열의 전부 또는 일부를 제거함으로써 생성될 수 있다. 페치 레코드들은 웹-크롤링 엔진 또는 페치 서비스에 의해 생성될 수 있다. 일부 구현들에서, 페치 레코드들은 예를 들어 인터넷을 통해 이용 가능한 문서들의 인덱싱 프로세스와 같은 인덱싱 프로세스의 일부로서 생성될 수 있다. 임베디드 리소스는 URL과 같은 식별자에 의해 페치 레코드 내에서 식별될 수 있다. 따라서, 웹페이지 또는 임베디드 리소스는 URL이라고도 지칭될 수 있으며, 임베디드 리소스(또는 웹페이지)와 URL에 대한 지칭은 본 명세서에 기재된 바와 같이 일반적으로 같은 의미라고 할 수 있다. 잠재적인 패턴은 쿼리 문자열의 전부 또는 일부를 제거함으로써 URL에 대해 생성될 수 있고, 쿼리 문자열은 임베디드 리소스에 대한 URL에서 물음표(?) 다음에 오는 임의의 문자들이다. 또한, 잠재적인 패턴은 서브도메인 이름을 제거 또는 무시하거나, 경로의 컴포넌트들(예를 들어, 첫 번째 슬래시("/")와 쿼리 문자열("?")사이의 컴포넌트들)을 무시하거나, 또는 쿼리 문자열의 특정 파라미터들을 무시함으로써 임베디드 리소스에 대해 생성될 수 있다. 일부 구현들에서, 페치 레코드들로부터의 임베디드 리소스는 이것의 URL로부터 생성된 하나보다 많은 잠재적인 패턴과 연관될 수 있다. 잠재적인 URL들은 분류(sorting)에 의해 또는 알려진 다른 클러스터링 기법들에 의해 클러스터링 될 수 있다.

[0029] 시스템은 잠재적인 패턴들 중 하나로 표현되는 클러스터를 선택할 수 있고(310), 클러스터 멤버들의 양이 임계치보다 크거나 같은지 여부를 결정할 수 있다(315). 클러스터 멤버들의 양은 이전 시간 기간 동안에 잠재적인 패턴과 매치하는 임베디드 리소스들이 페치된 횟수의 양을 나타낸다. 따라서, 하나의 특정한 임베디드 리소스가 한 번보다 많이 페치되었을 경우, 양에 있어서 여러 번 표현될 수 있다. 클러스터에 대한 양이 임계치 미만인 경우(315, 아니오), 시스템은 다음 클러스터(355)로 진행할 수 있다. 잠재적인 패턴이 충분한 양의 멤버들을 갖는 경우(315, 예), 시스템은 잠재적인 패턴과 매치하는 임베디드 리소스들의 샘플을 선택할 수 있다(320). 즉, 시스템은 클러스터의 고유 멤버들을 샘플링할 수 있다. 물론, 일부 구현들에서 시스템은 클러스터 내 모든 고유 임베디드 리소스들을 샘플로서 선택할 수 있다 (예를 들어, 샘플 사이즈가 100%). 그러면 시스템은 상기 샘플의 임베디드 리소스들 중 하나를 선택하고(325), 상기 선택된 임베디드 리소스로 리소스를 내장하는 웹페이지(예를 들어, 임베더 웹 페이지)의 제1 렌더링 결과를 생성할 수 있다(330). 임베더 웹 페이지는 페치 레코드들로부터 결정될 수 있다. 시스템은 또한, 상기 선택된 임베디드 리소스를 페치하지 않고 상기 임베더 웹페이지의 제2 렌더링 결과를 생성할 수 있다(330). 시스템은 예를 들어, 배치 렌더링 엔진을 사용하여 상기 두 개의 렌더링 결과들을 생성할 수 있다. 그 다음, 시스템은 상기 두 개의 렌더링 결과들이 유사한지의 여부를 결정하기 위해 상기 제1 렌더링 결과와 상기 제2 렌더링 결과를 비교할 수 있다(335).

[0030] 도 4와 관련하여 보다 상세하게 논의된 바와 같이, 시스템은 렌더링 결과들이 유사한지의 여부를 결정하기 위해 다양한 테스트들을 사용할 수 있다. 렌더링 결과들이 유사하지 않을 경우(335, 아니오), 시스템은 다음 클러스터로 이동할 수 있다(335). 렌더링 결과들이 유사할 경우(335, 예), 시스템은 클러스터 샘플 내에 검사할 다른 임베디드 리소스들이 있는지 결정할 수 있다(340). 만약 다른 임베디드 리소스들이 있는 경우(340, 아니오), 시스템은 상기 샘플에 대해 다음 임베디드 리소스를 선택하고, 두 개의 렌더링 결과들을 생성하고 결과들을 비교하는 단계 330과 335를 반복할 수 있다. 만약 샘플 내 모든 임베디드 리소스들이 검사되었을 경우(340, 예), 시스템은 상기 클러스터에 대한 잠재적인 패턴을 옵션적 리소스 패턴 데이터 스토어에 추가할 수 있다(350). 도 5와 관련하여 보다 상세하게 설명한 바와 같이, 이 데이터 스토어는 페치될 필요 없는 임베디드 리소스들을 식별하기 위해 서비스에 의해 사용될 수 있다. 그 다음, 시스템은 검사할 다른 클러스터들이 있는지 결정할 수 있다(355). 만약 분석할 다른 클러스터들이 남아 있는 경우(355, 아니오), 시스템은 다음 클러스터를 선택하고, 상기 다음 클러스터의 잠재적인 패턴과 매치하는 임베디드 리소스들에 대하여 단계 315 에서 355까지를 반복할 수 있다. 만약 분석할 클러스터가 더이상 없을 경우(355, 예), 프로세스(300)는 완료된다.

[0031] 예시 프로세스(300)는 옵션적이지 않은 하나의 임베디드 리소스가 옵션적 리소스 패턴 데이터 스토어에서 제외

될 잠재적인 패턴을 초래하는 구현을 나타낸다(예를 들어, 335, 아니오). 그러나, 일부 구현들에서 시스템은 옵션적인 잠재적 패턴에 대한 임베디드 리소스들의 리스트 및 요구되는 잠재적 패턴에 대한 임베디드 리소스의 리스트(예를 들어, 두 렌더링 결과들이 유사하지 않다고 고려됨)을 유지(keep)할 수 있다. 이러한 구현들에서, 시스템은 요구되는 잠재적 패턴과 매치하는 임베디드 리소스들의 백분율을 계산할 수 있다. 상기 백분율이 매우 낮을 때(예를 들어, 1% 이하), 시스템은 여전히 옵션적 리소스 패턴 데이터 스토어에 상기 잠재적 패턴을 추가할 수 있다(350). 따라서, 구현들은 도 3에 도시된 정확한 프로세스(300)에 국한되지 않는다.

[0032] 일부 구현들에서, 시스템은 서로 다른 브라우져 유형에 의해 렌더링된 렌더링 결과들로 프로세스(300)의 일부 또는 전부를 반복할 수 있다. 예를 들어, 브라우저 유형은 풀 브라우저 또는 모바일 브라우저가 될 수 있다. 제1 및 제2 렌더링 결과가 풀 브라우저에 의해 렌더링된 경우, 시스템은 모바일 브라우저를 사용하여 단계 330에서 350까지 반복할 수 있다. 따라서, 옵션적 리소스 패턴들의 리스트는 브라우저 유형의 표시를 포함할 수 있고, 데이터 스토어 내 패턴들은 또한 브라우저 유형의 표시를 포함할 수 있다. 이는 시스템이 임베디드 리소스가 하나의 브라우저 유형(예를 들어, 모바일 브라우저)에 대하여 옵션적인 것이고 다른 브라우저 유형(예를 들어, 풀 브라우저)에 대하여는 옵션적이지 않은 것인지를 표시할 수 있게 한다.

[0033] 도 4는 구현에 따른, 두 개의 렌더링 결과들이 유사한지를 결정하기 위한 예시적인 프로세스(400)를 도시하는 흐름도이다. 프로세스(400)는 예를 들어, 도 3의 단계(335)의 일부로서 수행될 수 있다. 프로세스(400)가 렌더링 결과의 유사성을 결정하기 위한 세 가지 테스트들을 도시하지만, 구현들은 도시된 테스트들 중 하나, 둘 또는 모든 테스트들을 포함할 수 있음이 이해된다.

[0034] 프로세스(400)는 두 개의 렌더링 결과들 내 토큰들을 비교하는 것을 포함할 수 있다. 토큰들은 예를 들면, 사용자가 볼 수 있는 단어들과 같은 문서의 텍스트를 포함한다. 토큰들은 도 2의 DOM 트리(210)와 같은 DOM 트리를 처리함으로써 생성될 수 있다. 일부 구현들에서, 비교를 수행하기 전에 정지(stop) 단어들과 숫자들이 제거될 수 있다. 제1 렌더링 결과에 대한 토큰들과 제2 렌더링 결과에 대한 토큰들이 다른 경우(410, 예), 시스템은 스크립팅이 켜져있는 렌더링에 의해 추가된 고유 토큰들의 양이 토큰 임계치보다 작은지의 여부를 옵션적으로 결정할 수 있다(415). 즉, 스크립팅이 켜진 렌더링이 오직 몇 개의 고유 토큰들을 초래할 경우, 시스템은 그 차이가 유의미하지 않은 것이라 간주할 수 있다. 일부 구현들에서 토큰 임계치는 5가 될 수 있다. 추가된 고유 토큰들의 양이 토큰 임계치를 만족할 경우(415, 아니오), 시스템은 렌더링 결과들이 유사하지 않은 것으로 여길 수 있다(430). 렌더링 결과들이 유사하지 않을 때, 상기 렌더링 결과들을 생성하기 위해 사용된 임베디드 리소스는 옵션적인 것이 아니다. 일부 구현들에서, 시스템은 테스트(415)를 스킵할 수 있고, 토큰들의 임의의 차이들이 모두 유의미하고 렌더링 결과들은 유사하지 않은 것으로 여겨질 수 있다. 양이 토큰 임계치를 만족하지 못할 경우(415, 예), 시스템은 토큰들의 차이가 유의미하지 않다고 여길 수 있다. 따라서, 제1 렌더링 결과에 대한 고유 토큰들이 제2 렌더링 결과에 대한 토큰들과 동일하거나(410, 아니오) 또는 고유 토큰들의 양이 토큰 임계치를 만족하지 못하는 경우, 상기 렌더링 결과들은 유사한 것으로 여겨지며, 시스템은 다른 테스트들을 계속해서 수행할 수 있다. 토큰 테스트만을 포함하는 구현들에서, 시스템은 렌더링 결과들이 유사하다는 결정과 함께 바로 단계(445)로 진행할 수 있다.

[0035] 일부 구현들에서, 토큰들이 같을 경우(410, 아니오) 또는 (415, 예), 시스템은 아웃링크들(420)을 비교할 수 있다. 아웃링크들은 렌더링된 페이지에서 다른 웹 페이지들 또는 다른 문서들로의 링크들을 나타낸다. 아웃링크들은 또한, 렌더링 결과의 DOM 트리의 앵커 태그 노드들(<a> 태그들)로부터 추출될 수 있다. 시스템이 아웃링크들에서 차이를 발견할 경우(425, 예), 시스템은 상기 차이가 유의미하고 렌더링 결과들은 유사하지 않은 것으로 여길 수 있다(430). 따라서 임베디드 리소스가 옵션적인 것으로 여겨지지 않는다. 두 개의 렌더링 결과들의 아웃링크들이 같을 경우(425, 아니오), 시스템은 상기 렌더링 결과들이 유사한 것으로 여길 수 있거나(예를 들어, 단계(450)로 진행함으로써) 또는 계속해서 다른 유사성 테스트들을 수행할 수 있다. 일부 구현들에서, 시스템은 단계(405)와 단계(415) 이전에 그리고 이와는 독립적으로 단계(420)와 단계(425)를 수행할 수 있다.

[0036] 기타 테스트들을 대신하거나 또는 추가적으로 수행될 수 있는 또 다른 유사성 테스트에서, 시스템은 이미지(예를 들어, 스냅샷), DOM 트리, 또는 레이아웃 사이의 유사성을 결정할 수 있다. 예를 들어, 시스템은 DOM 트리, 레이아웃, 이미지 또는 렌더링 결과의 또 다른 컴포넌트에 대한 최장 공통 시퀀스(LCS)를 계산할 수 있다(435). 시스템은 동일한 컴포넌트의 백분율을 나타내는 유사성 점수를 계산하기 위해 LCS를 사용할 수 있다(440). 상기 백분율이 임계치를 만족할 때(445, 예), 시스템은 렌더링 결과들이 유사하다고 판단할 수 있다(450). 상기 백분율이 임계치를 만족하지 못할 때(445, 아니오), 시스템은 렌더링 결과들이 유사하지 않다고 판단할 수 있다. 일단 시스템이 렌더링 결과들이 유사한지(450) 또는 유사하지 않은지(430)에 대하여 결정하면 프로세스(400)는 종

료한다.

[0037] 물론, 시스템은 프로세스(400)의 일부로서 다른 테스트들을 수행할 수 있다. 예를 들어, 또 다른 유사성 테스트에서(도 4에 나타나지 않음) 시스템이 레이아웃의 주된 컴포넌트들(major component)을 결정할 수 있다. 렌더링 결과의 레이아웃은 박스들로 구성되며 각각의 박스는 화면 좌표들로 정의된다. 일반적으로, 각 박스는 웹페이지의 각각의 요소에 대응한다. 예를 들어, (비록 모든 DOM요소들이 대응하는 렌더 박스를 가질 수 있는 것은 아니지만) 레이아웃은 DOM 트리에 있는 DOM 노드들의 박스표현들을 포함할 수 있다. 박스들은 렌더 트리로서 또한 알려진 트리 구조로 조직화된다. 따라서, 예를 들어 표는 레이아웃에 있는 박스로 나타내어 질 수 있고, 문단은 레이아웃에 있는 다른 박스로 표현될 수 있다. 웹페이지의 주된 컴포넌트들은 화면 좌표들로써 정의된 바와 같이, 가장 큰 박스들이 있는 레이아웃에 있는 요소들이다. 일부 구현들에서, 예를 들어 세트가 가장 큰 주된 컴포넌트를 포함하도록 시스템은 상기 세트에 미리 결정된 양의 주된 컴포넌트들을 포함시킬 수 있다. 일부 구현들에서, 시스템은 세트에 있는 화면의 백분율을 구성하는 주된 컴포넌트들을 포함할 수 있다. 이러한 구현에서 가장 큰 주된 컴포넌트가 백분율보다 많이 차지하는 경우, 상기 가장 큰 주된 컴포넌트는 세트의 유일한 멤버가 될 수 있다. 일부 구현들에서, 임계 크기를 초과하는 박스 크기를 갖는 임의의 컴포넌트는 주된 컴포넌트들의 세트에 포함될 수 있다. 주된 컴포넌트들의 세트에 포함되지 않은 컴포넌트는 보조 컴포넌트(minor component)로 간주 될 수 있다. 일부 구현들에서, 시스템은 주된 컴포넌트를 찾기 위해 어니언 필링 기법(onion-peeling technique)을 사용할 수 있다. 예를 들어, 렌더 트리의 루트 박스에서 시작하여, 시스템은 너비 우선 탐색(breath-first search)을 수행할 수 있고 루트 박스의 가장 큰 차일드 박스를 식별할 수 있다. 그 다음, 시스템은 가장 큰 차일드를 선택할 수 있고, 현재 박스의 가장 큰 차일드 박스(예를 들어, 루트박스의 가장 큰 차일드 박스)를 찾기 위해 더 깊이 갈 수 있다. 시스템은 차일드 박스들 중 아무것도, 예를 들어 페어런트 박스의 면적의 절반 이상을 차지하는 것과 같이, 지배적이지 않을 때 더 깊숙하게 가는 것을 중단할 수 있다. 시스템이 더 깊숙하게 가는 것을 중단할 때, 주된 컴포넌트는 지배적인 차일드를 가지지 못한 박스가 된다.

[0038] 시스템은 세트 내 주된 컴포넌트들 간 유사성 점수를 계산할 수 있다. 예를 들어, 시스템은 주된 컴포넌트 박스들의 오버랩 점수를 사용할 수 있다. 오버랩 점수에서 시스템은 주된 컴포넌트의 중첩된(overlapping) 영역을 계산할 수 있는데, 이는 제2 렌더링 결과에서 대응하는 주된 컴포넌트의 영역과 중첩되는 제1 렌더링 결과의 주된 컴포넌트의 영역을 나타낸다. 그 다음, 시스템은 각 주된 컴포넌트의 총 면적에 대해 중첩된 영역의 조화평균(harmonic average)을 나타내는 유사성 점수를 계산할 수 있다. 예를 들어, 시스템은 다음과 같은 식을 사용할 수 있다. $\frac{2}{(a1/oa)+(a2/oa)}$ 여기서, oa는 중첩된 영역이고, a1은 제1 렌더링 결과 내 주된 컴포넌트 박스의 총 면적, a2는 제2 렌더링 결과 내 주된 컴포넌트 박스의 총 면적이다. 물론, 시스템은 점수 계산을 위해 다른 유사성 매트릭들을 사용할 수 있다. 이러한 유사성 매트릭들의 예시들은 카츠(Katz) 유사성을 포함하지만, 이에 국한되지 않는다. 유사성 점수가 유사성 임계치를 만족할 경우, 렌더링 결과들은 유사한 것으로 간주 되므로 임베디드 리소스는 옵션적이다. 일부 구현들에서 유사성 임계치는 높을 수 있으며, 예를 들면 80% 이상일 수 있다. 유사성 점수가 임계치를 만족하지 못 할 경우, 시스템은 렌더링 결과들이 유사하지 않은 것으로 간주할 수 있다. 일부 구현들에서, 유사성 점수가 유사 임계치를 만족하는 경우, 시스템은 도 4에 도시되지 않은 렌더링 결과들에 기초하여, 추가적인 유사성 테스트들을 수행할 수 있다. 일부 구현들에서, 시스템은 또한, 보조 컴포넌트들에서의 차이들에 대하여 유사성 점수를 계산할 수 있으나, 이러한 차이들에 훨씬 낮은 가중을 둘 수 있는 바, 이는 예를 들어, 이런 차이들을 렌더링 결과들에 대한 최종 유사성 점수의 아주 작은 부분이 되게 한다.

[0039] 일부 구현들에서, 시스템은 프로세스(400)에서 비결정론(indeterminism)의 영향들을 최소화하거나 제거하려는 시도를 할 수 있다. 비결정론은 임베디드 리소스들이 동일하더라도 렌더링 결과가 상이할 때 발생한다. 비결정론은 옵션적인 임베디드 리소스들이 마치 요구되는 것처럼 보이게 할 수 있다. 따라서, 비결정론을 고려하는 것은 옵션적 리소스들의 범위를 상당히 증가시킬 수 있다. 비결정론을 고려하기 위하여, 시스템은 제3 렌더링 결과를 생성할 수 있다. 제3 렌더링 결과는 (예를 들어, 샘플 URL을 포함하는) 제1 렌더링 결과의 생성으로부터 반환된 리소스들을 사용하는 렌더링일 수 있다. 제3 렌더링 결과와 제1 렌더링 결과 간에서 발생한 모든 차이들은 비결정론 때문이다. 시스템은 비결정론에 기인한 차이들(예를 들어, 제1 렌더링 결과와 제3 렌더링 결과에서 서로 다른 아웃링크들, 토큰들, 이미지, 렌더 트리 등에서의 차이들)을 모두 제거함으로써 유사성을 세니타이징(sanitizing)할 수 있다. 예를 들어, 시스템은 제1 렌더링 결과의 이미지와 제3 렌더링 결과의 이미지 간에 다른 픽셀들의 수를 나타내는 제1 양과, 제1 렌더링 결과와 제2 렌더링 결과의 이미지들 간에 다른 픽셀들의 수를 나타내는 제2 양을 계산할 수 있다. 시스템은 차이를 산출하기 위해 상기 제1 양을 상기 제2 양에서 뺄 수 있다. 차이가 상기 제2 양과 가까울수록 상기 제1 렌더링 결과와 상기 제2 렌더링 결과 간의 차이들은 비결정론

에 의한 것이 아닐 가능성이 높아진다. 따라서, 시스템은 차이에 기초하여 LCS의 유사성 점수를 조정할 수 있다(예를 들어, 차이가 제2 양과 같을 경우 조정이 필요 없으며, 차이가 제2 양의 절반 일 경우 절반을 조정한다). 다른 예로서, 시스템은 제1 렌더링 결과와 제3 렌더링 결과 간에 다른 DOM 노드들을 결정할 수 있으며, 제1 렌더링 결과와 제2 렌더링 결과를 비교할 때(예를 들어, LCS를 계산할 때) 이러한 노드들은 무시할 수 있다.

[0040] 도 5는 구현에 따른, 최적화된 렌더링에 대한 정보를 제공하는 예시적 프로세스(500)을 도시한 흐름도이다. 프로세스(500)는 브라우저에 대한 서비스로서 실행될 수 있다. 브라우저는 도 1의 브라우저(140)와 같은, 모바일 또는 풀인 클라이언트 브라우저일 수 있거나, 또는 도 1의 배치 렌더링 엔진들과 같은 배치 렌더링 엔진일 수 있다. 일부 구현들에서 서비스는 클라우드 기반 서비스일 수 있다. 즉, 브라우저는 네트워크를 통해, 브라우저에 응답을 제공할 수 있는 클라우드 기반 서비스로 요청을 전송할 수 있다. 일부 구현들에서, 서비스는 도 1의 배치 렌더링 시스템(110)과 같은, 서버상에서 실행될 수 있다. 또 다른 구현들에서, 서비스는 브라우저가 실행되는 컴퓨팅 디바이스에 국부적일 수 있다. 예를 들어, 옵션적 리소스 패턴들의 데이터 스토어는 브라우저를 실행하는 컴퓨팅 디바이스에 푸쉬될 수 있으며, 서비스는 컴퓨팅 디바이스에서 실행될 수 있다. 일부 구현들에서 서비스는 도 1의 클라이언트(180)와 같은 클라이언트 상에서 실행될 수 있다.

[0041] 프로세스(500)는 서비스가 브라우저로부터 URL을 수신하는 것으로 시작한다(505). 브라우저는 모바일 브라우저 또는 풀 브라우저 일 수 있으며, 클라이언트 또는 배치 렌더링 엔진 상에서 실행될 수 있다. 서비스는 요청으로부터의 URL이 옵션적 리소스 패턴과 매치하는지 여부를 결정할 수 있다(510). 일부 구현들에서, 요청된 URL 전체에서 매치가 이루어지도록 패턴은 전체 URL을 나타낼 수 있다. 일부 구현들에서 패턴은, 예를 들어 쿼리 문자열이 제거된 또는 다양한 부분들이 와일드 카드들로 대체된, 부분적인 URL을 나타낼 수 있다. 요청된 URL이 옵션적 리소스 데이터 스토어에서 적어도 하나의 패턴과 매치할 경우(501, 예), 서비스는 상기 URL이 옵션적인 것임을 나타내는 응답을 제공하고, 상기 임베더 웹페이지는 컨텐츠를 페치하지 않고 렌더링 될 수 있다(515). 일부 구현들에서, 이는 리소스를 찾을 수 없을 때 브라우저가 수신할 수 있는, 응답 "URL 미발견(URL not found)"와 유사한 응답일 수 있다. 이 표시를 수신한 것에 응답하여, 상기 브라우저는 상기 임베디드 리소스를 스킵하는 임베더 웹페이지를 렌더링할 수 있고, 이는 프로세싱과 대역폭 리소스들을 절약한다. 요청된 URL이 옵션적 리소스 데이터 스토어의 패턴과 매치하지 않으면(510, 아니오), 서비스는 임베디드 리소스가 요구됨을 나타내는 응답을 제공한다(520). 일부 구현들에서, 서비스는 실제로 페치를 수행하고 상기 임베디드 리소스에 대한 컨텐츠를 제공할 수 있다. 일부 구현들에서, 서비스는 브라우저가 상기 임베디드 리소스에 대한 페치 요청을 진행하도록 지시하는 응답을 제공할 수 있다. 따라서, 브라우저는 상기 임베디드 리소스에 대한 컨텐츠로 임베더 웹페이지를 렌더링할 수 있다. 그 다음, 프로세스(500)는 브라우저가 렌더링 프로세스를 최적화하는데 사용하는 정보를 제공하면서 종료한다. 브라우저는 서비스에 브라우저 유형을 제공할 수 있고, 서비스는 옵션적 리소스 데이터 스토어 내 패턴들을 매치시킬 때 상기 브라우저 타입을 사용할 수 있다.

[0042] 도 6은 도 1의 배치 렌더링 시스템(110) 및/또는 클라이언트(180)로써 작동되는 일반적인 컴퓨터 디바이스(600)의 예를 도시하며, 이는 본 명세서에 설명된 기법들과 함께 사용될 수 있다. 컴퓨팅 디바이스(600)는 랩탑들, 데스크탑들, 워크스테이션들, 개인정보 단말기들, 휴대전화들, 스마트폰들, 타블렛들, 서버들, 그리고 웨어러블 디바이스들을 포함하는 기타 컴퓨팅 디바이스들과 같은, 컴퓨팅 디바이스들의 다양한 예시적 형태들을 나타내기 위한 것이다. 본 명세서에 설명된 컴포넌트들과 그 연결들, 관계들, 그리고 기능들은 예시적인 것들일 뿐이며 본 명세서에서 묘사 및/또는 청구되는 발명들의 구현들을 제한하기 위함이 아니다.

[0043] 컴퓨팅 디바이스(600)는 예를 들어, 인터페이스(608)를 통해 연결되는 실리콘-기반 하드웨어 프로세서 예컨대, 프로세서(602), 메모리(604), 저장 디바이스(606), 그리고 확장 포트들(610)을 포함한다. 일부 구현들에서, 컴퓨팅 디바이스(600)는 다른 컴포넌트들 중에서도, 인터페이스(608)를 통해 연결되는 송수신기(646), 통신 인터페이스(644), 그리고 GPS(Global Positioning System) 수신기 모듈(648)을 포함할 수 있다. 디바이스(600)는 필요에 따라 디지털 신호처리 회로를 포함할 수 있는 통신 인터페이스(644)를 통해 무선으로 통신할 수 있다. 컴포넌트들(602, 604, 606, 608, 610, 640, 644, 646 및 648)은 공통 마더보드에, 또는 다른 적절한 방식들로 장착될 수 있다.

[0044] 프로세서(602)는 디스플레이(616)와 같은 외부 입/출력 디바이스 상의 GUI를 위한 그래픽 정보를 디스플레이하기 위해, 메모리(604)에 또는 저장 디바이스(606) 상에 저장된 명령어들을 포함하여 컴퓨팅 디바이스(600) 내에서의 실행을 위한 명령어들을 프로세스 할 수 있다. 디스플레이(616)는 모니터 또는 플랫 터치스크린 디스플레이일 수 있다. 일부 구현들에서, 다수의 메모리들 및 메모리 유형들과 함께 다수의 프로세서들 및/또는 다수의 버스들이 적절하게 사용될 수 있다. 또한, 다수의 컴퓨팅 디바이스들(600)이 연결될 수 있으며, 각각의 디바이스는 (예를 들어, 서버 뱅크, 블레이드 서버들의 그룹 또는 멀티프로세서 시스템으로서) 필요한 작업들의 일부

를 제공한다.

[0045] 메모리(604)는 컴퓨팅 디바이스(600) 내의 정보를 저장한다. 한 구현에서, 메모리(604)는 휘발성 메모리 유닛 또는 유닛들이다. 다른 구현에서 메모리(604)는 비휘발성 메모리 유닛 또는 유닛들이다. 메모리(604)는 또한 자기 또는 광학 디스크와 같은 컴퓨터 판독가능 매체의 또 다른 형태일 수 있다. 일부 구현들에서, 메모리(604)는 확장 인터페이스를 통해 제공되는 확장 메모리를 포함할 수 있다.

[0046] 저장 디바이스(606)는 컴퓨팅 디바이스(600)에 대해 대용량 스토리지(storage)를 제공할 수 있다. 한 구현에서, 저장 디바이스(606)는 스토리지 영역 네트워크 또는 다른 구성들을 포함하여 컴퓨터 판독가능 매체, 예를 들어 플로피디스크 디바이스, 하드디스크 디바이스, 광디스크 디바이스 또는 테이프 디바이스, 플래시메모리 디바이스 또는 다른 유사한 고체 상태 메모리 디바이스, 또는 디바이스들의 어레이일 수 있거나 이를 포함할 수 있다. 컴퓨터 프로그램 물은 컴퓨터 판독가능 매체에 유형으로 수록될 수 있다. 컴퓨터 프로그램 물은 또한 명령어들을 포함할 수 있는 바, 상기 명령어들은 실행될 때, 상술 된 방법들과 같은 하나 이상의 방법들을 수행한다. 컴퓨터 또는 기계 판독가능 매체는 메모리(604), 저장 디바이스(606), 또는 프로세서(602) 상의 메모리와 같은 저장 디바이스이다.

[0047] 인터페이스(608)는 컴퓨팅 디바이스(600)에 대한 대역폭-집약적 동작들(bandwidth-intensive operations)을 관리하는 고속제어기, 또는 저 대역폭-집약적 동작들을 관리하는 저속제어기, 또는 이러한 제어기들의 조합일 수 있다. 외부 인터페이스(640)는 디바이스(600)가 다른 디바이스들과의 근거리 통신을 가능하게 하기 위해 제공될 수 있다. 일부 구현들에서, 제어기(608)는 저장 디바이스(606) 및 확장 포트(614)와 결합 될 수 있다. 다양한 통신포트들(예를 들어, USB, 블루투스, 이더넷, 무선 이더넷(Ethernet))을 포함할 수 있는 확장 포트는, 키보드, 포인팅 디바이스, 스캐너, 또는 스위치나 라우터와 같은 네트워킹 디바이스와 같은 하나 이상의 입/출력 디바이스들에 (예를 들어, 네트워크 어답터를 통하여) 결합될 수 있다.

[0048] 컴퓨팅 디바이스(600)는 도면에 도시된 바와 같이 다양한 형태들로 구현될 수 있다. 예를 들어, 이는 표준서버(630)로써 구현되거나 또는 그러한 서버들의 그룹에서 여러 번 구현될 수 있다. 또한, 이는 랙 서버시스템의 일부로서 구현될 수도 있다. 게다가, 이는 랩 탑 컴퓨터(632), 데스크 탑 컴퓨터(634) 또는 타블렛이나 스마트폰(636)과 같은 개인용 컴퓨팅 디바이스에서 구현될 수도 있다. 전체 시스템은 서로 통신하는 다수의 컴퓨팅 디바이스들(600)로 구성될 수 있다. 다른 구성들도 가능하다.

[0049] 도 7은 본 명세서에 설명된 기법들과 함께 사용될 수 있는, 도 1의 시스템(110)일 수 있는 일반적인 컴퓨터 디바이스(700)의 한 예시를 나타낸다. 컴퓨팅 디바이스(700)는 서버들, 블레이드 서버들, 데이터 센터들, 메인 프레임들 및 다른 대규모 컴퓨팅 디바이스들과 같은 대규모 데이터 처리 디바이스들의 다양한 예시적 형태들을 나타내기 위한 것이다. 컴퓨팅 디바이스(700)은 가능하게는 하나 이상의 통신 네트워크들에 의해 상호접속된 네트워크 어태치 스토리지 노드들을 포함하는 다수의 프로세서들을 가진 분산 시스템일 수 있다. 여기에 표시된 컴포넌트들, 그리고 그들의 연결들, 관계들, 그리고 기능들은 예시들일 뿐이며, 본 명세서에 묘사 및/또는 청구되는 발명들의 구현들을 제한하기 위함이 아니다.

[0050] 분산 컴퓨팅 디바이스(700)는 임의의 수의 컴퓨팅 디바이스들(780)을 포함할 수 있다. 컴퓨팅 디바이스(780)는 로컬 또는 광역 네트워크, 전용 광 링크들, 모뎀들, 브릿지들, 라우터들, 스위치들, 유선 또는 무선 네트워크들 등을 통해 통신하는 서버 또는 렉 서버들, 메인프레임들 등을 포함할 수 있다.

[0051] 일부 구현들에서 각 컴퓨팅 디바이스는 다수의 렉들을 포함할 수 있다. 예를 들어, 컴퓨팅 디바이스(780a)는 다수의 렉들(758a-758n)을 포함할 수 있다. 각각의 렉은 프로세서들(752a-752n 및 762a-762n)과 같은, 하나 이상의 프로세서들을 포함할 수 있다. 프로세서들은 데이터 프로세서들, 네트워크 어태치 저장 디바이스(network attached storage devices)들 및 기타 컴퓨터 제어 디바이스들을 포함할 수 있다. 일부 구현들에서, 한 프로세서는 마스터 프로세서로서 동작할 수 있고, 스케쥴링 및 데이터 분배작업들을 제어할 수 있다. 프로세서들은 하나 이상의 렉 스위치들(758)을 통해 상호 접속될 수 있고, 하나 이상의 렉들은 스위치(778)를 통해 접속될 수 있다. 스위치(778)는 다수의 접속된 컴퓨팅 디바이스들(700) 간의 통신을 처리할 수 있다.

[0052] 각각의 렉은 메모리(754) 및 메모리(764)와 같은 메모리와, 스토리지(756) 및 스토리지(766)와 같은 스토리지(storage)를 포함할 수 있다. 스토리지(756 및 766)는 대용량 스토리지를 제공할 수 있고, 스토리지 영역 네트워크(Storage area network) 내 디바이스들 또는 기타 구성들을 포함하는, 네트워크-결합 디스크들(network-attached disks), 플로피 디스크들, 하드디스크들, 광 디스크들, 테이프들, 플래시 메모리 또는 다른 유사한 고체상태 메모리 디바이스들, 또는 디바이스들의 어레이와 같은, 휘발성 또는 비휘발성 스토리지를 포함할 수 있

다. 스토리지(756 또는 766)는 다수의 프로세서들, 다수의 렉들, 또는 다수의 컴퓨팅 디바이스들 간에 공유될 수 있고, 하나 이상의 프로세서들에 의해 실행 가능한 명령어들을 저장하는 컴퓨터 판독가능 매체를 포함할 수 있다. 메모리(754 및 764)는 예를 들어, 휘발성 메모리 유닛 또는 유닛들, 비휘발성 메모리 유닛 또는 유닛들, 및/또는 자기 또는 광디스크들, 플레시 메모리, 캐시, RAM(Random Access Memory), ROM(Read Only Memory), 그리고 이들의 조합들과 같은, 컴퓨터 판독가능 매체의 기타 형태들을 포함할 수 있다. 메모리(754)와 같은 메모리는 또한, 프로세서들 (752a-752n) 간에 공유될 수 있다. 인덱스와 같은 데이터구조들은 예를 들어, 스토리지(756) 및 메모리(754)에 걸쳐 저장될 수 있다. 컴퓨팅 디바이스(700)는 도시되지 않은 제어기, 버스, 입/출력 장치, 통신 모듈 등과 같은 기타 컴포넌트들을 포함할 수 있다.

[0053] 시스템 110과 같은 전체 시스템은 서로 통신하는 다수의 컴퓨팅 디바이스들(700)로 구성될 수 있다. 예를 들어, 디바이스(780a)는 디바이스들(780b), (780c), 그리고 (780d)와 통신할 수 있으며, 이들은 총괄적으로 시스템(100)이라고도 할 수 있다. 또 다른 예시로, 도 1의 시스템(110)은 하나 이상의 컴퓨팅 디바이스들(700)을 포함할 수 있다. 일부 컴퓨팅 디바이스들은 지리적으로 서로 가깝게 배치될 수 있으며, 다른 컴퓨팅 디바이스들은 지리적으로 멀리 위치될 수 있다. 컴퓨팅 디바이스(700)의 레이아웃은 한 예시일 뿐이며, 시스템은 다른 레이아웃들 또는 구성들을 취할 수 있다.

[0054] 다양한 구현들은, 저장 시스템, 적어도 하나의 입력 디바이스 및 적어도 하나의 출력 디바이스로부터 데이터 및 명령어들을 수신하고 그리고 이들에 데이터 및 명령어들을 전송하기 위해 결합된, 특수용 또는 범용일 수 있는, 기판에 형성된 적어도 하나의 프로그램가능 프로세서를 포함하는 프로그램가능 시스템 상에서 실행가능 및/또는 해석가능한 하나 이상의 컴퓨터 프로그램들의 구현을 포함할 수 있다.

[0055] 이들 컴퓨터 프로그램들(프로그램들, 소프트웨어, 소프트웨어 어플리케이션들 또는 코드로도 알려짐)은 프로그램 가능 프로세서에 대한 기계 명령어들을 포함하고, 하이-레벨 절차(high-level procedural) 및/또는 객체-지향(object-oriented) 프로그래밍 언어 및/또는 어셈블리/기계언어로 구현될 수 있다. 본 명세서에 기재된 "기계 판독가능 매체","컴퓨터 판독가능 매체"라는 용어는 기계 명령어들 및/또는 데이터를 프로그램가능 프로세서에 제공하는데 사용되는 임의의 비-일시적인 컴퓨터 프로그램 물, 장치 및/또는 디바이스(예를 들어, 자기디스크, 광디스크, 메모리(RAM 포함), PLD(Programmable Logic Devices))를 지칭한다.

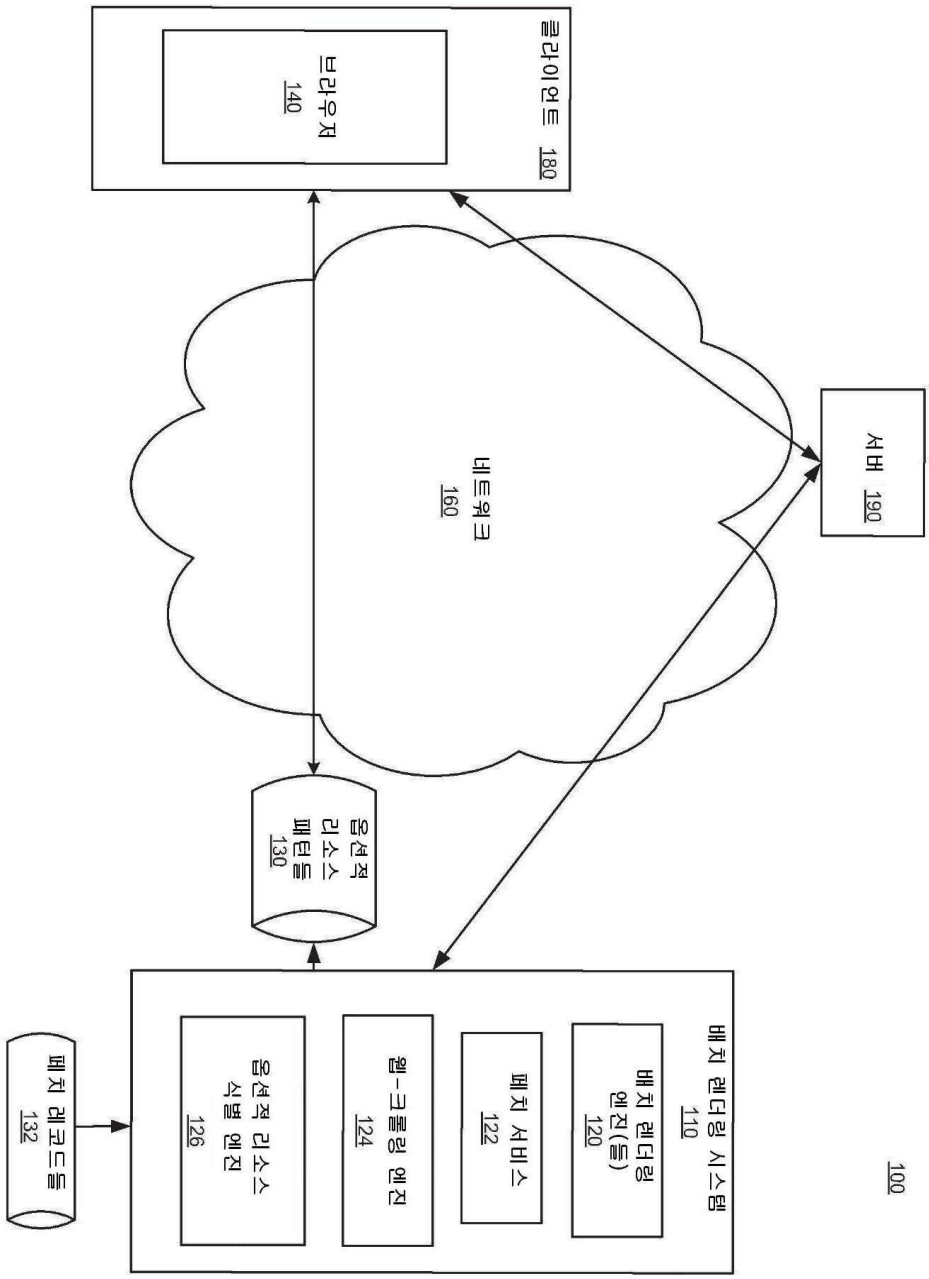
[0056] 본 명세서에서 설명된 시스템들과 기법들은 백 엔드 컴포넌트(예를 들어, 데이터 서버로서)를 포함하거나, 미들웨어 컴포넌트(예를 들어, 어플리케이션 서버)를 포함하거나, 프론트 엔드 컴포넌트(예를 들어, 사용자가 본 명세서에 설명된 시스템들 및 기법들의 구현과 상호작용할 수 있게 하는 그래픽 사용자 인터페이스 또는 웹 브라우저를 갖는 클라이언트 컴퓨터) 또는 이러한 백 엔드, 미들웨어나 프론드 엔드 컴포넌트들의 임의의 조합을 포함하는 컴퓨팅 시스템에서 구현될 수 있다. 시스템의 컴포넌트들은 디지털 데이터 통신(예를 들어, 통신 네트워크)의 임의의 형태 또는 매체에 의해 상호접속될 수 있다. 통신 네트워크들의 예시들은 근거리 통신망("LAN"), 광역 통신망("WAN"), 그리고 인터넷을 포함한다.

[0057] 컴퓨팅 시스템은 클라이언트들과 서버들을 포함한다. 클라이언트와 서버는 일반적으로 서로로부터 원격이며, 전형적으로 통신 네트워크를 통해 상호작용한다. 클라이언트 및 서버의 관계는, 각각의 컴퓨터들 상에서 실행되고 서로에 대하여 클라이언트-서버 관계를 가지는 컴퓨터 프로그램들에 의해 발생된다.

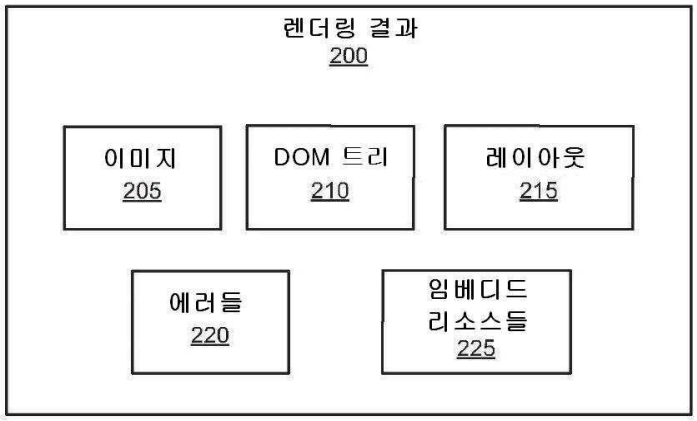
[0058] 다수의 구현들이 설명되었다. 그럼에도 불구하고, 본 발명의 사상 및 범위를 벗어나지 않고 다양한 변형이 이루어질 수 있다. 또한, 도면들에서 묘사된 논리 흐름들은 바람직한 결과들을 달성하기 위해 도시된 특정 순서 또는 순차적 순서를 요구하지 않는다. 또한, 다른 단계들이 제공되거나 설명된 흐름 들로부터 단계들이 제거될 수 있으며, 다른 컴포넌트들이 설명된 시스템들에 추가되거나 상기 설명된 시스템으로부터 제거될 수 있다. 따라서, 다른 구현들은 다음의 특허청구범위 내에 있다.

도면

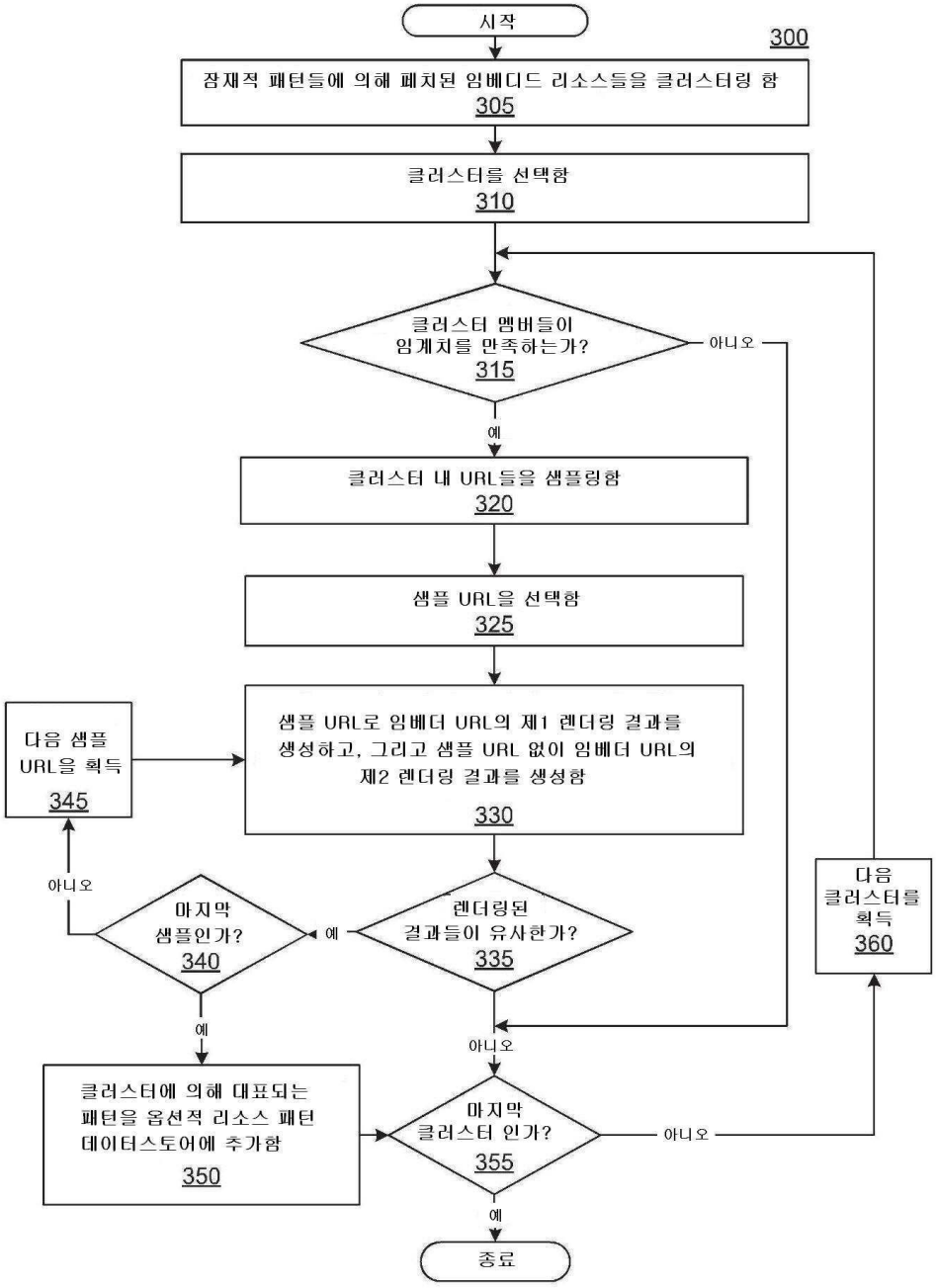
도면1


클라이언트 180
브라우저 140
네트워크 160
서버 190
옵션적 리소스 패턴들 130
배치 렌더링 시스템 110
배치 렌더링 엔진(들) 120
페지 서비스 122
웹-크롤링 엔진 124
옵션적 리소스 식별 엔진 126
페지 레코드들 132
100

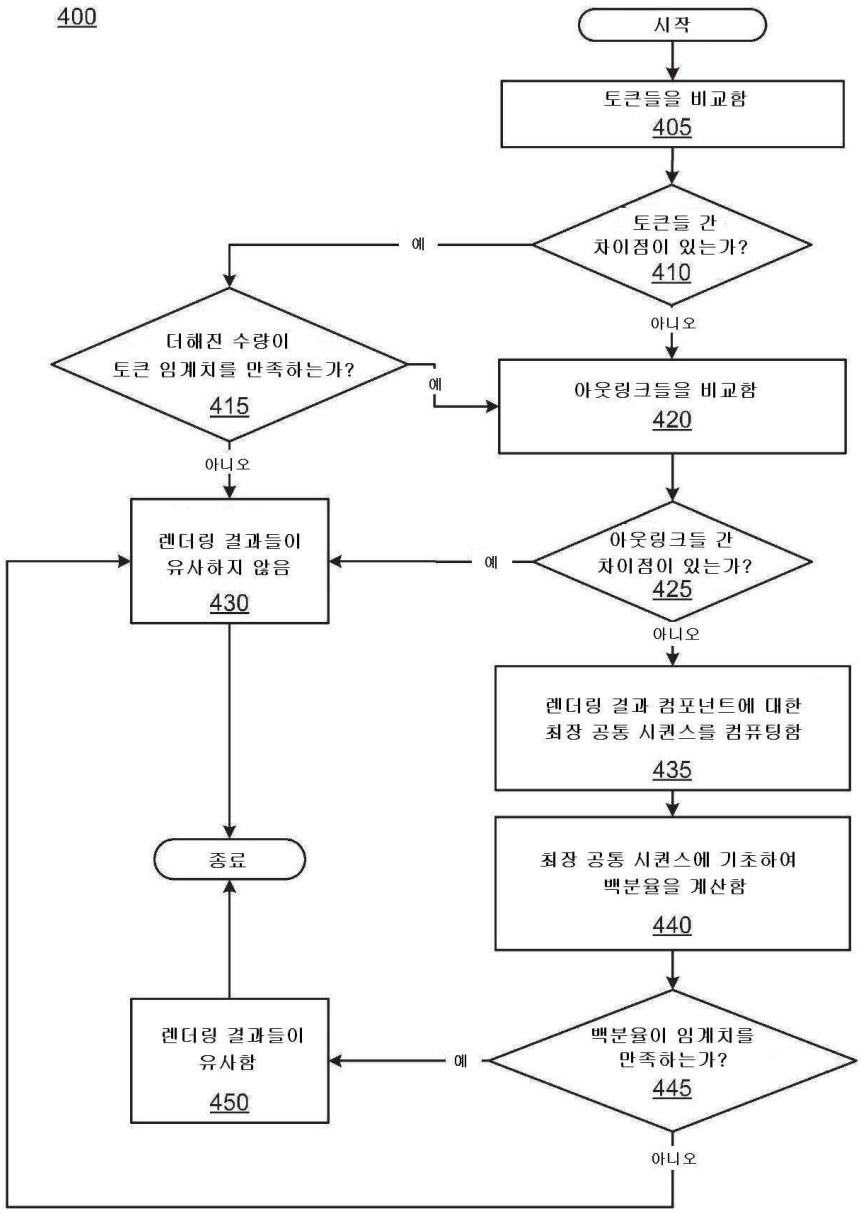
도면2



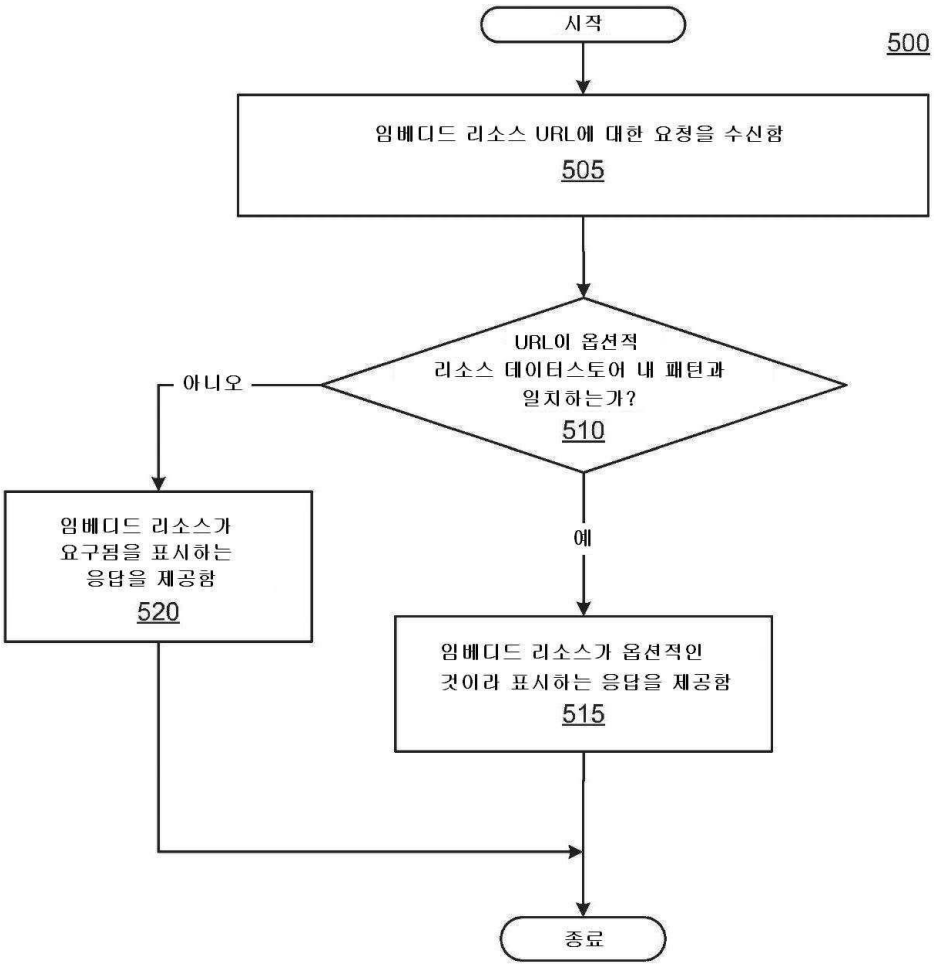
도면3


시작
300
잠재적 패턴들에 의해 페치된 임베디드 리소스들을 클러스터링 함
305
클러스터를 선택함
310
클러스터 멤버들이 임계치를 만족하는가?
315
아니오
예
클러스터 내 URL들을 샘플링함
320
샘플 URL을 선택함
325
샘플 URL로 임베더 URL의 제1 렌더링 결과를 생성하고, 그리고 샘플 URL 없이 임베더 URL의 제2 렌더링 결과를 생성함
330
다음 샘플 URL을 획득
345
아니오
마지막 샘플인가?
340
예
렌더링된 결과들이 유사한가?
335
아니오
다음 클러스터를 획득
360
예
클러스터에 의해 대표되는 패턴을 옵션적 리소스 패턴 데이터스토어에 추가함
350
마지막 클러스터 인가?
355
아니오
예
종료

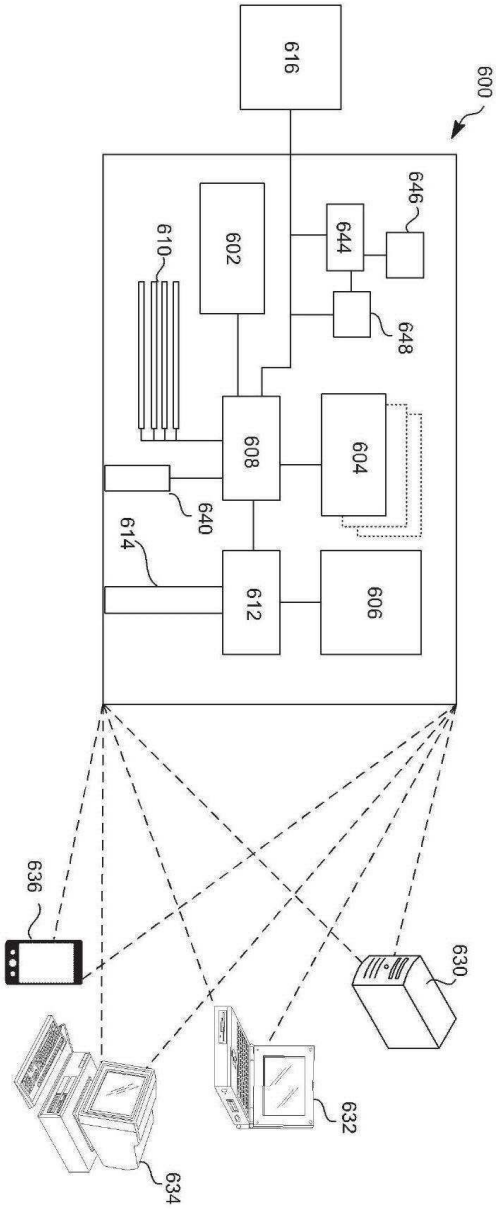
도면4


400
시작
토큰들을 비교함
405
토큰들 간 차이점이 있는가?
410
예
아니오
더해진 수량이 토큰 임계치를 만족하는가?
415
예
아니오
아웃링크들을 비교함
420
렌더링 결과들이 유사하지 않음
430
아웃링크들 간 차이점이 있는가?
425
예
아니오
렌더링 결과 컴포넌트에 대한 최장 공통 시퀀스를 컴퓨팅함
435
종료
최장 공통 시퀀스에 기초하여 백분율을 계산함
440
렌더링 결과들이 유사함
450
백분율이 임계치를 만족하는가?
445
예
아니오

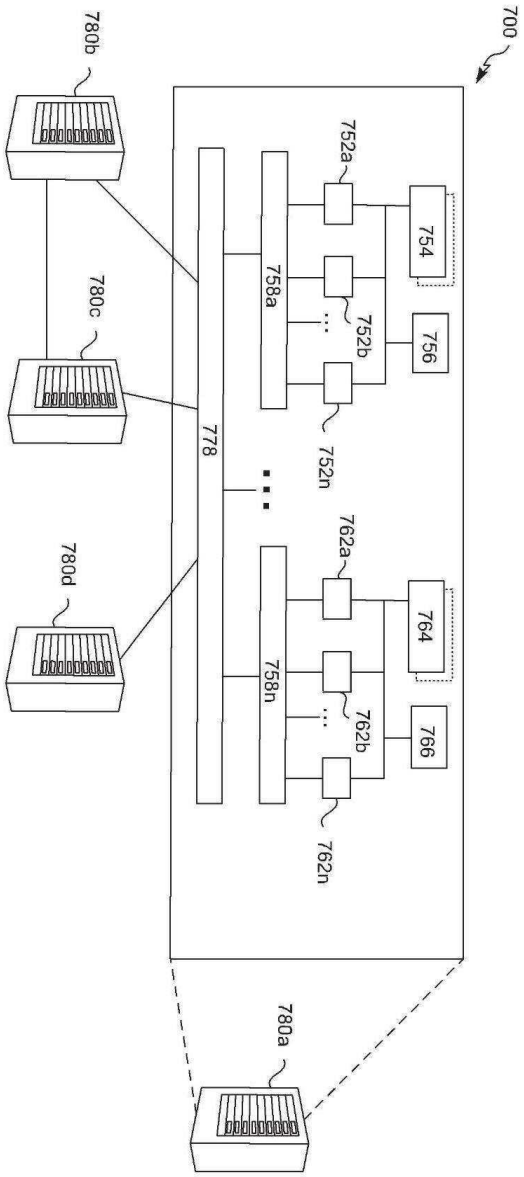
도면5


시작
500
임베디드 리소스 URL에 대한 요청을 수신함
505
URL이 옵션적
리소스 데이터스토어 내 패턴과
일치하는가?
510
아니오
예
임베디드 리소스가
요구됨을 표시하는
응답을 제공함
520
임베디드 리소스가 옵션적인
것이라 표시하는 응답을 제공함
515
종료

도면6


616
600
646
644
602
610
648
608
604
640
614
612
606
636
630
632
634

도면7


700
780b
780c
780d
780a
778
758a
758n
752a
752b
752n
754
756
762a
762b
762n
764
766