Open Community Runtime (OCR)

Intel OCR Team

21st Feb 2017

Notices

Acknowledgment: This material is based upon work supported by Lawrence Livermore National Labs subcontract B608115.

Disclosure Notice: This presentation is bound by Non-Disclosure Agreements between Intel Corporation, the Department of Energy, and DOE National Labs, and is therefore for Internal Use Only and not for distribution outside these organizations or publication outside this Subcontract.

USG Disclaimer: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Intel Disclaimer: Intel makes available this document and the information contained herein in furtherance of DesignForward, FastForward and the Extreme Scale Initiative. None of the information contained therein is, or should be construed, as advice. While Intel makes every effort to present accurate and reliable information, Intel does not guarantee the accuracy, completeness, efficacy, or timeliness of such information. Use of such information is voluntary, and reliance on it should only be undertaken after an independent review by qualified experts.

Access to this document is with the understanding that Intel is not engaged in rendering advise or other professional services. Information in this document may be changed or updated without notice by Intel.

This document contains copyright information, the terms of which must be observed and followed.

Reference herein to any specific commercial product, process or service does not constitute or imply endorsement, recommendation, or favoring by Intel or the US Government.

Intel makes no representations whatsoever about this document or the information contained herein. IN NO EVENT WILL INTEL BE LIABLE TO ANY PARTY FOR ANY DIRECT, INDIRECT, SPECIAL OR OTHER CONSEQUENTIAL DAMAGES FOR ANY USE OF THIS DOCUMENT, INCLUDING, WITHOUT LIMITATION, ANY LOST PROFITS, BUSINESS INTERRUPTION, OR OTHERWISE, EVEN IF INTEL IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

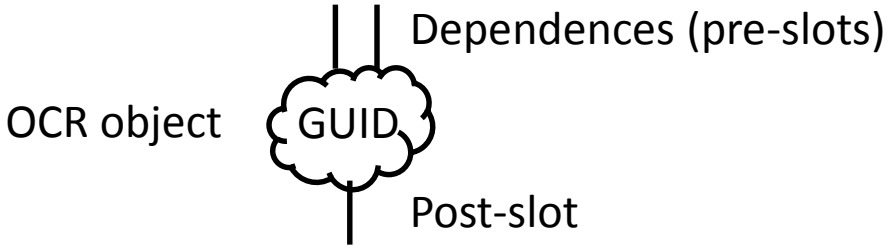
Overview

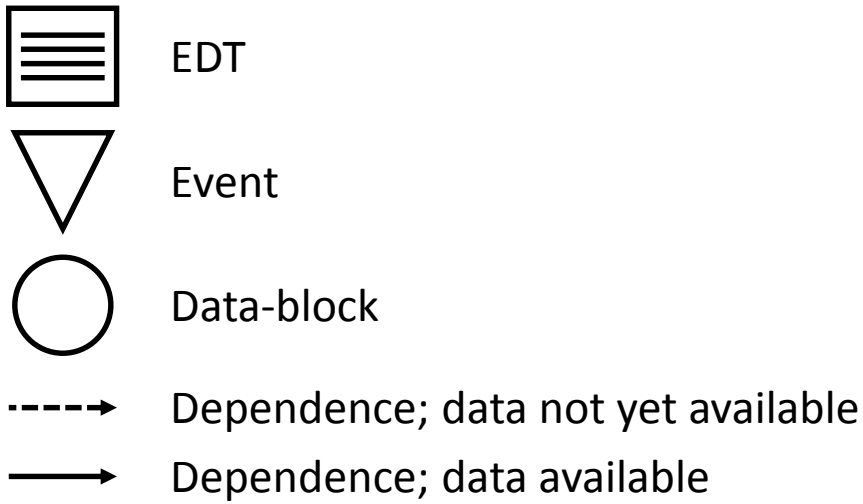
- OCR – Background
- API
- Requirements driven by application usage scenarios
- Requirements driven by platform

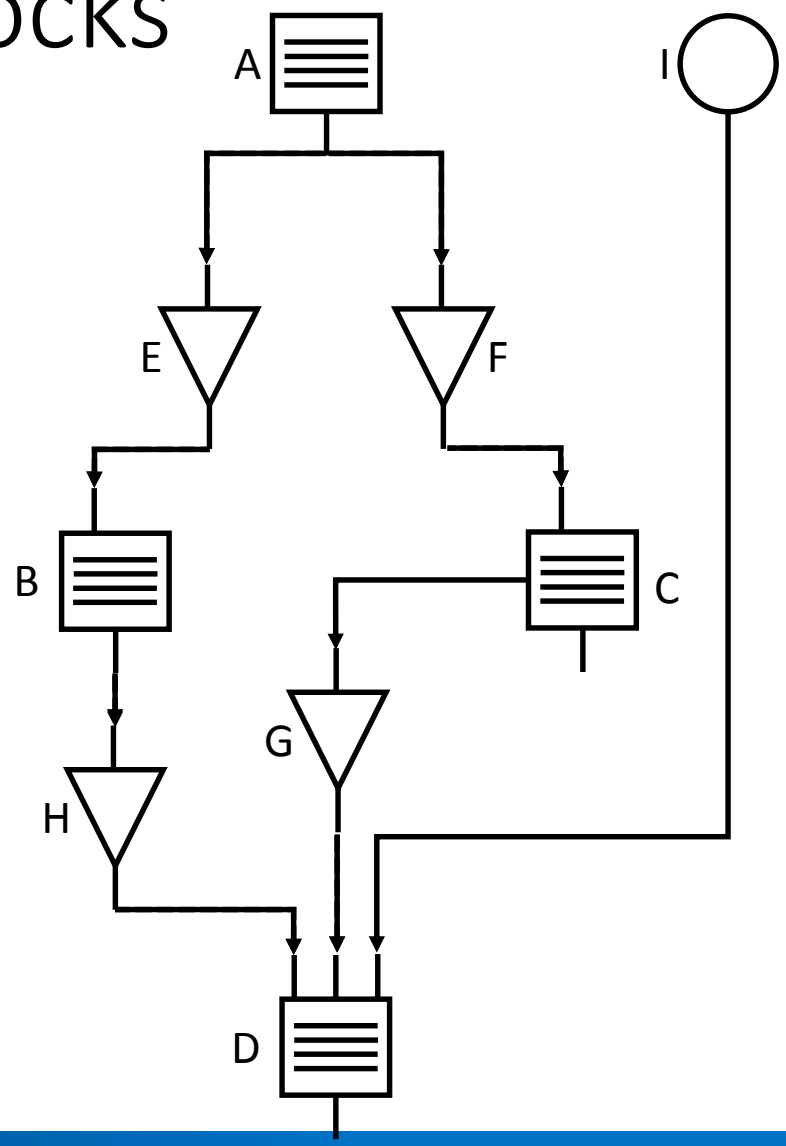
OCR - Introduction

- Asynchronous event-driven task-based runtime
- Explicit data-dependence
- Low-level API defined in a community developed specification
- Support for
  - higher-level libraries/runtimes
  - introspection, dynamic adaptation and resiliency

OCR Building Blocks







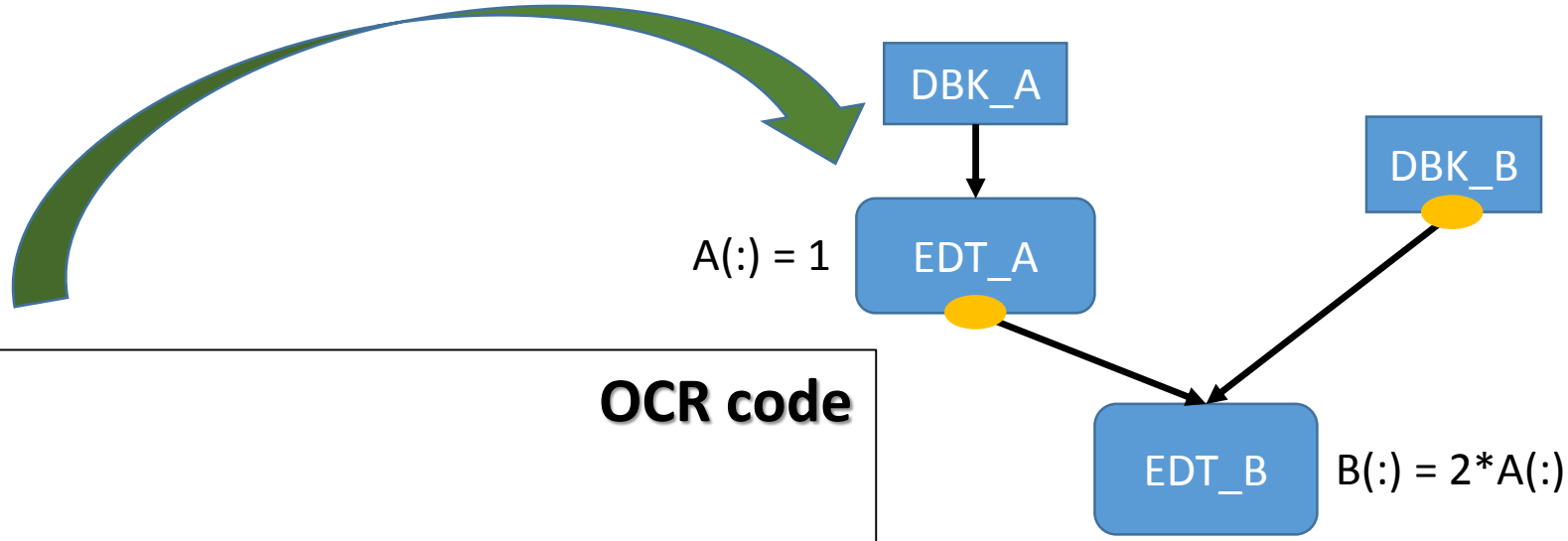
Dynamically build a Directed Acyclic Graph (DAG) of the program

OCR Deep-Dive: A Simple Task-Graph

```
double* PTR_A; //malloc(size);
double* PTR_B; //malloc(size);
```

Plain C code

```
computeA( PTR_A ); //A(:)=1;
computeB( PTR_B);  //B(:)=2*A(:)
```



OCR code

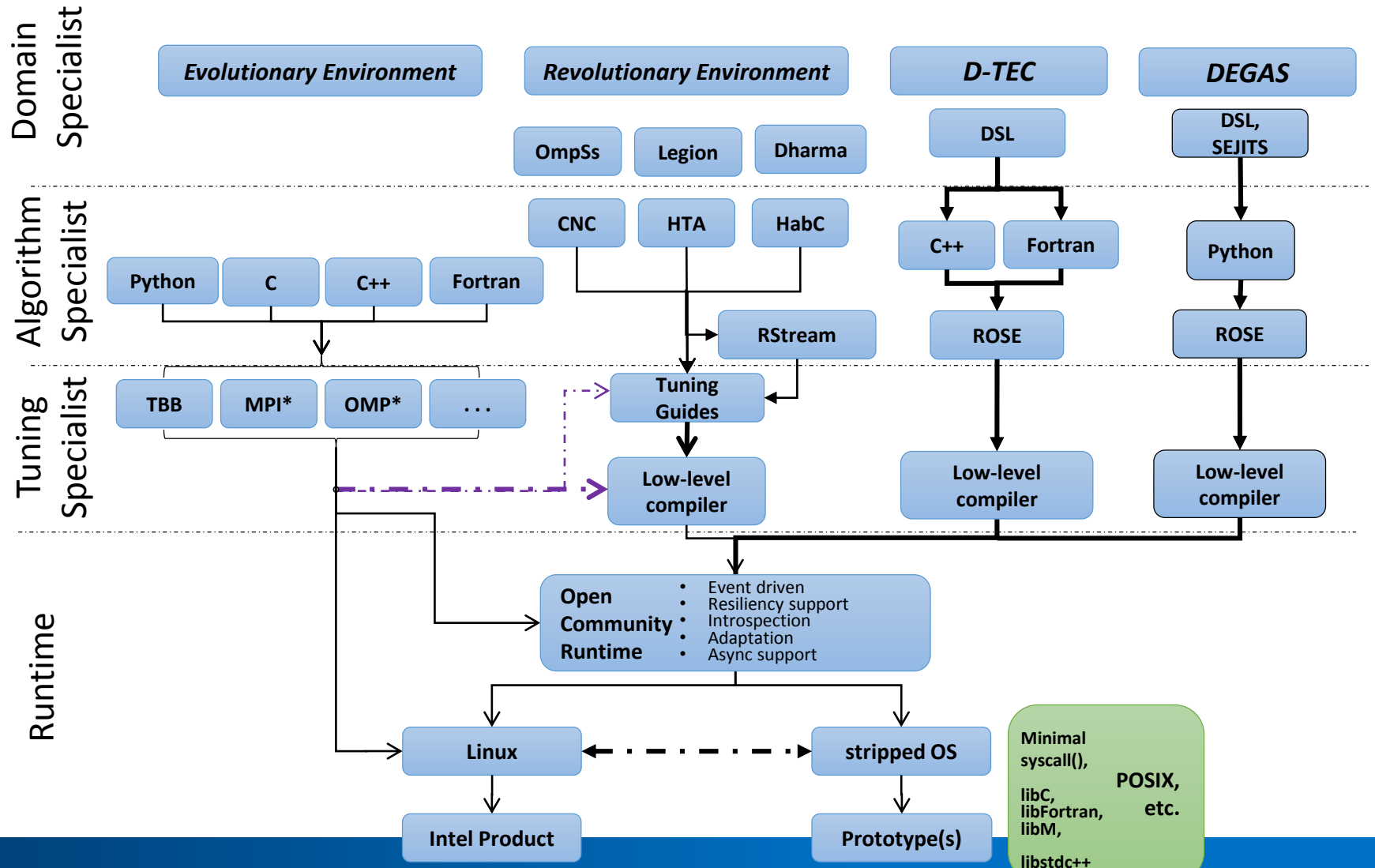
```
ocrGuid_t DBK_A; //ocrDbCreate();
ocrGuid_t DBK_B; //ocrDbCreate()

ocrEdtCreate(&EDT_A, computeA,…, &EVT_A); //1 dependence
ocrEdtCreate(&EDT_B, computeB,…, &EVT_B); //2 dependences

ocrAddDependence( DBK_A, EDT_A, 0, DB_MODE_RW);       //EDT_A dep: 0

ocrAddDepencence( EVT_A, EDT_B, 0, DB_MODE_RO);       //EDT_B dep: 0
ocrAddDependence( DBK_B, EDT_B, 1, DB_MODE_RW);       //EDT_B dep: 1
```

OCR Ecosystem



Loss of Semantic Information

Need for semantic information (tuning, hints, guides) (collectives, affinity, stream, ..)

Platforms Supported

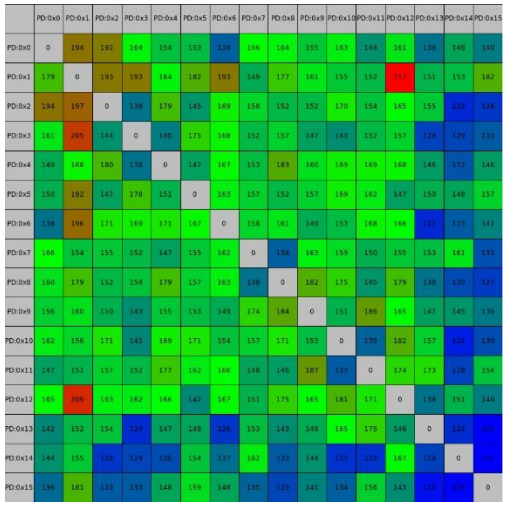
- x86 single node
- x86 distributed
  - MPI-based
  - GASNet-based
- Xeon-Phi (KNL)
- Exascale strawman (Traleika Glacier) architecture
  - Simulator
  - FPGA emulator (ongoing)
  - Full-chip (expected mid-2017)

High-level Language Support

- OCxxR (C++) - Rice University
- R-Stream – Reservoir Labs
- Legion/Realm – Stanford/Rice University
- OmpSs – Barcelona Supercomputing Center
- CnC (Concurrent Collections) – Rice University
- HCLib – Rice University
- HTA – UIUC

Productivity

- MPI-Lite (allows MPI programs to use OCR without any code changes)
- Autogen (generate OCR from simple Python description)
- AutoOCR (generate OCR from annotations to C code)
- Libraries
  - SPMD-like expressions
  - Reduction patterns
- Tools
  - EDT visualization
  - Communication “heat-map”
  - Debug: runtime profiling, tracing



Implementations

- Community-driven implementation
  - https://xstack.exascale-tech.com
- University of Vienna
  - Dokulil, J. et al., “Implementing the Open Community Runtime for Shared-Memory and Distributed-Memory Systems”, PDP 2016
  - Dokulil, J. et al., “Retargeting of the Open Community Runtime to Intel Xeon Phi”, ICCS 2015
- Pacific Northwest National Lab
  - Landwehr, J. et al., “Application characterization at scale: lessons learned from developing a distributed open community runtime system for high performance computing”, Computation & Communication Frontiers 2016

API

- EDT
  - **Task templates**: *ocrEdtTemplateCreate(), ocrEdtTemplateDestroy()*
  - **Tasks**: *ocrEdtCreate(), ocrEdtDestroy()*
- DBs
  - **Datablock management**: *ocrDbCreate(), ocrDbDestroy()*
  - **Datablock usage**: *ocrDbRelease()*
- Events
  - **Event management**: *ocrEventCreate(), ocrEventDestroy()*
  - **Event satisfaction**: *ocrEventSatisfy()*
  - **Dependence definition**: *ocrAddDependence()*
- Miscellaneous
  - **Entry point of OCR**: *mainEdt()*
  - **Shutdown**: *ocrShutdown()*

API Extensions

- Hints
  - Generic methods to pass application related information down to runtime
  - No enforcement guarantees
- Affinity
  - Associates an affinity handle with an object handle
  - Allows for platform-specific interpretation
- Labeling
  - Associates a label string to a GUID
- Pause/Resume

Requirements from Applications (1/2)

- Legacy application migration support
  - MPI-driven design is ubiquitous and lots of work have gone into them
  - Native SPMD support
    - Creating & forking 1M+ asynch tasks
    - Efficient point-to-point messaging & collectives
    - DAG creation with global knowledge (without state having to be passed down tasks)
    - Possibly via a library
- Template extension
  - Richer expression other than just single tasks
  - Reusable iteration DAGs

Requirements from Applications (2/2)

- Tasks
  - Flexibility in granularity; task hierarchy
- Data
  - PGAS addressing interface with hierarchical datablock implementation
  - Runtime decides best layout – AOS vs. SOA
  - Runtime decides best granularity (based on hints & performance)
  - Support for striding, scatter/gather access patterns
  - Native support for multi-dimensional arrays

Requirements from Platform

- Resiliency
  - Handle data corruption, core failure with minimal interruption
- Dynamic adaptation
  - Continuous performance-driven introspection and adaptation
  - Load balancing
- Native support for specific features, acceleration
  - Collectives, scatter/gather, etc.
- Heterogeneity

Q & A