Parallelization of the Fitting

Updates

X.Valls for the ROOT Team

23/02/2017

Workers in TProcessExecutor

- **Problem**: TProcessExecutor was creating too many workers.

- **Solution**: Extending the chunking interface to TProcessExecutor. (PR #364)

Streaming of VecCore types

- **Problem**: We could not stream LikelihoodAux<Double_v> because it contained VecCore data members.

Streaming of VecCore types

- Solution:

```
#ifdef __CLING__
            double logvalue[vecCore::VectorSize<T>()];
            double weight[vecCore::VectorSize<T>()];
            double weight2[vecCore::VectorSize<T>()];
#else
            T logvalue{};
            T weight{};
            T weight2{};
#endif
        };
```

MultiProcessing of the fitting

- **Problem**: Works and it's consistent, but evaluation of the Likelihood differs from TThreadExecutor and sequential cases.

- **Idea**: Try with Kahan compensate summation algorithm.

Times: Multithreading (4 thr)

	NCALLS	TIME (s)	SPEEDUP	Theor. MAX SPEEDUP
Sequential	160	19.9	1	1
Parallel	159	6.11	3.25	4
SSE2	159	10.71	1.86	2
SSE2 Parallel	139	2.75	7.24	8

	NCALLS	TIME (s)	SPEEDUP	Theor. MAX SPEEDUP
Sequential	160	19.49	1	1
AVX2	160	5.58	3.49	4
AVX2 Parallel	139	1.52	12.82	16

Times: Multiprocessing

2NCores workers, just finished running.

	NCALLS	TIME (s)	SPEEDUP	Theor. MAX SPEEDUP
Sequential	160	19.92	**1**	1
Multiprocess	137	8.25	**2.41**	4
AVX2	160	5.61	**3.55**	4
AVX2 MP	137	4.99	**3.99**	16

What's next

- Study of the chunking in TProcessExecutor.
- Time Chi2 with multiprocessing.
- Add a function in ROOT for the compensate sum (Kahan Summation Algorithm) and compare results.
- Continue with the integration in ROOT(currently in PR #1-#2, waiting for VecCore in ROOT for the next one).
- Improve tests, get code ready for GSoC students.