Update on UrQMD Production and Reading

Fan Si

2025/04/22

Overview

- Source: ui:/ustcfs/HICUser/fsi/urqmd_250421.tar.gz
- Location: ui:/ustcfs/HICUser/fsi/urqmd
- Subdirectories
  - urqmd (original UrQMD production code)
  - curqmd (this production code)
  - run (script for production)
  - UrqmdDst (classes for storage)
  - UrqmdUtilities (new since current version)
  - PdgData (new since current version)
- C++98 is supported (PdgData requires C++11)

Introduction: $Location/urqmd

- Taken from https://itp.uni-frankfurt.de/~bleicher/index.html?content=urqmd
- Change: pythia6409.f: –> pytahi6428.f (final PYTHIA6, https://pythia.org/)
- Change: mk/Linux.mk: new gfortran option -std=legacy
  - Otherwise unable to be compiled with high versions of gfortran
- Change: GNUmakefile: pythia6409 –> pytahi6428

Introduction: $Location/curqmd

- For UrQMD production and its storage in ROOT Tree

- Contents:
  - main.{cxx,h}: C++ –> (Fortran <–> C++)
  - c_%.f: modified from $Location/urqmd/%.f
  - GNUmakefile
    - Calls $Location/urqmd/GNUmakefile
    - Calls $Location/UrqmdDst/GNUmakefile
    - Compiles c_*.f and main.cxx
    - Links $Location/urqmd/*.o (only those different from $Location/curqmd/c_*.o), $Location/curqmd/{c_*.o,main.o} to executable main

Introduction: $Location/run

- Contents:
  - in.urqmd: input configuration of UrQMD production
  - random.C: ROOT code for random number generation to random.list
    - Input: n (random number counter, required), name (file name for existing numbers, default="")
    - Output: file random.list containing n different random numbers (beginning with existing ones)
      - 1<= random number <= maximum of 4-byte integer
    - Used for UrQMD input seed for random number generator
    - Able to read existing numbers since current version
  - run.bash: script for calling UrQMD production
    - Input: index (serial number for the job)
    - Sets the seed as the index-th number in random.list and saves the updated in.urqmd to in/${index}.urqmd
    - Output: out/${index}.root (event IDs begin with ${index}*(#events per job))

Introduction: $Location/run

- Contents:
  - run.condor: condor configuration template
  - run_ui.condor: condor configuration template with specific settings for ui
  - submit.bash: script for submitting condor jobs using run.condor
    - Input: number (number of jobs, default=100)
    - Saves the updated run.condor to script/run_${number}.condor
    - Log files (error, log, output) saved in log/job_$(Process).{err,log,out}
  - resubmit.bash: script for resubmitting condor jobs
    - Input: condor configuration file (e.g. script/run_${number}.condor)
    - Resubmits jobs with errors or incomplete jobs

Introduction: $Location/UrqmdDst

- Classes for storage of UrQMD data

- Contents:
  - UrqmdDst.{cxx,h}: saves pointers to sub-structures and statuses of them
  - UrqmdEvent.{cxx,h}: saves event information
  - UrqmdColllision.{cxx,h}: saves collision information
  - UrqmdTrack.{cxx,h}: saves track information
  - UrqmdTrackEx.{cxx,h}: saves track extended information
  - GNUmakefile
    - Compiles code in this directory to (lib)UrqmdDst.so

Introduction: class UrqmdDst

- Useful functions:
  - UrqmdEvent* event()
  - UrqmdTimestep* timestep(UInt_t i)
  - UrqmdCollision* collision(UInt_t i)
  - UrqmdTrack* track(UInt_t i)
  - UrqmdTrackEx* trackEx(UInt_t i)
  - UInt_t numberOfTimesteps()
  - UInt_t numberOfCollisions()
  - UInt_t numberOfTracks()
  - TChain* initInputChain(TString inputFileName, const TString inputTreeName = "urqmd")
    - Reads UrqmdDst tree from file (*.root or *.list)
  - TTree* initOutputTree(const TString outputTreeName = "urqmd")
    - Creates UrqmdDst tree for writing

Introduction: class UrqmdEvent

- Useful functions:
  - UInt_t id()
    - Serial number of event
  - Double_t b(): impact parameter

Introduction: class UrqmdTimestep

- Useful functions:
  - Double_t time(): time of this timestep
  - UInt_t nPart(): not defined by UrQMD
    - Calculated by (total # projectile and target nucleons) − (# tracks with parentCollisionType%100==0)
    - parentCollisionType%100==0 means particle did not go through inelastic scattering
  - UInt_t nColl(): nElasticColl+nInelasticColl+nBlockedColl
  - UInt_t nElasticColl(), UInt_t nInelasticColl(), UInt_t nBlockedColl()
  - UInt_t nDecays(), UInt_t nHardRes(), UInt_t nSoftRes(), UInt_t nDecayRes()
  - UInt_t nTracks(): number of tracks at this timestep
  - UInt_t startTrack(): "nTracks" tracks starting from track index "startTrack" belong to this timestep

- If CTOption(4)==1, original UrQMD outputs initial configuration (t==0)
  - This code saves timestep at t==0

Introduction: class UrqmdTimestep

```
if ( (itypt(1).eq.ityp(inew(i))).and.
     (iso3t(1).eq.iso3(inew(i))) ) then
   origin(inew(i))=origint(1)+100
   uid(inew(i))=uidt(1)
elseif ( (itypt(2).eq.ityp(inew(i))).and.
        (iso3t(2).eq.iso3(inew(i))) ) then
   origin(inew(i))=origint(2)+100
   uid(inew(i))=uidt(2)
```

- Useful functions:
  - Double_t time(): time of this timestep
  - UInt_t nPart(): not defined by UrQMD
    - Calculated by (total # projectile and target nucleons) − (# tracks with parentCollisionType%100==0)
    - parentCollisionType%100==0 means particle did not go through inelastic scattering
  - UInt_t nColl(): nElasticColl+nInelasticColl+nBlockedColl
  - UInt_t nElasticColl(), UInt_t nInelasticColl(), UInt_t nBlockedColl()
  - UInt_t nDecays(), UInt_t nHardRes(), UInt_t nSoftRes(), UInt_t nDecayRes()
  - UInt_t nTracks(): number of tracks at this timestep
  - UInt_t startTrack(): "nTracks" tracks starting from track timestep
- If CTOption(4)==1, original UrQMD outputs initial
  - This code saves timestep at t==0

column#		contents
0		"E" (only in o16)
1	√	# of collisions
2	√	# of elastic collisions
3	√	# of inelastic collisions
4	√	# of Pauli-blocked collisions
5	√	# of decays
6	√	# of produced *hard* baryon resonances
7	√	# of produced *soft* baryon resonances
8	√	# of baryon resonances produced via a decay of another resonance

Introduction: class UrqmdCollision

- (Serial number (starting from 1) not saved, equal to index; 0th collision always empty)

column#		format	contents
1	√	(i8)	number of ingoing particles N_{in}
2	√	(i8)	number of outgoing particles N_{out}
3	√	(i4)	process ID (see Table 11)
4		(i7)	collision/entry counter
5	√	(f8.3)	collision time t_{coll} in fm/c
6	√	(e12.4)	center of mass energy of the collision $\sqrt{s}$ in GeV
7	√	(e12.4)	total cross-section of the collision σ_{tot} in mbarn
8	√	(e12.4)	partial cross-section of the actual sub-process σ_{i} in mbarn
9	√	(e12.4)	Baryon density at collision point ρ_{B} in units of ρ_0

- Useful functions:
  - Double_t time(): time of this colllision
  - Double_t sqrtS(): $\sqrt{s}$ of this collision
  - Double_t sigmaTotal(): total cross section (mbarn)
  - Double_t sigmaPartial(): partial cross section (mbarn) of actual subprocess
  - Double_t rhoB(): baryon density (in units of ρ_0) at collision point
  - UInt_t type(): type index of process defined by UrQMD
  - UInt_t nIn(): number of incident particles in this collision
  - UInt_t nOut(): number of outgoing particles in this collision
  - UInt_t nTracks(): nIn+nOut
  - UInt_t startTrack(): "nTracks" tracks starting from track index "startTrack" belong to this collision; first "nIn" tracks are incident, last "nOut" tracks are outgoing

Introduction: class UrqmdTrack

- Useful functions:
  - UInt_t id(): serial number (not particle type ID)
  - UInt_t timestep()
    - If timestep==0 and collision!=0, this track belongs to that collision
  - UInt_t collision(): collision==0 should be ignored
    - If collision==0 (even if timestep==0), this track belongs to that timestep
  - Int_t iType(): type index according to isospin defined by UrQMD
  - Int_t i3(): double 3rd component of isospin
  - Int_t q(): double charge
  - Int_t s(): double spin (not saved by original UrQMD, may be ineffective)
  - Double_t m(): mass (in UrQMD, may not be equal to that in PDG)
  - Double_t x(), Double_t y(), Double_t z()
  - Double_t px(), Double_t py(), Double_t pz()

Introduction: class UrqmdTrack

- Useful functions:
  - UInt_t id(): serial number (not particle type ID)
  - UInt_t timestep()
    - If timestep==0 and collision!=0, this track belongs to that coll
  - UInt_t collision(): collision==0 should be ignored
    - If collision==0 (even if timestep==0), this track belongs to tha
  - Int_t iType(): type index according to isospin defined by
  - Int_t i3(): double 3rd component of isospin
  - Int_t q(): double charge
  - Int_t s(): double spin (not saved by original UrQMD, may
  - Double_t m(): mass (in UrQMD, may not be equal to that
  - Double_t x(), Double_t y(), Double_t z()
  - Double_t px(), Double_t py(), Double_t pz()

o13	o14	o15	o16	contents
		1		ind: index of particle (see CTOption(56))
1	1	2	1	t: time of particle
2	2	3	2 √	r_x: x coordinate
3	3	4	3 √	r_y: y coordinate
4	4	5	4 √	r_z: z coordinate
5	5	6	5	E: energy of particle
6	6	7	6 √	p_x: x momentum component
7	7	8	7 √	p_y: y momentum component
8	8	9	8 √	p_z: z momentum component
9	9	10	9 √	m: mass of particle
10	10	11	10 √	ityp: particle-ID
11	11	12	11 √	iso3: $2 \cdot I_3$ (see Section 1.2)
12	12	13	12 √	ch : charge of particle
13	13	14	13	parent collision number (see Table 10)
14	14	15	14	N_{coll} number of collisions
		16		S : strangeness
15	15		15	parent process type (see Table 11)
		17		history information (debugging only)
16				t^{fr} : freeze-out time of particle
17				r_x^{fr} : freeze-out x coordinate
18				r_y^{fr} : freeze-out y coordinate
19				r_z^{fr} : freeze-out z coordinate
20				E^{fr} : freeze-out energy of particle
21				p_x^{fr} : freeze-out momentum x component
22				p_y^{fr} : freeze-out momentum y component
23				p_z^{fr} : freeze-out momentum z component
	16*			τ_{dec} decay time of particle
	17*			τ_{form} formation time of particle
	18*			R_σ cross section reduction factor
	19*		√	unique particle number (not ID!)
			16*	$\mathrm{ityp}_1^{\mathrm{old}}$: particle-ID of parent particle # 1
			17*	$\mathrm{ityp}_2^{\mathrm{old}}$: particle-ID of parent particle # 2

Introduction: class UrqmdTrack

- Useful functions:
  - UInt_t nCollisions()
  - UInt_t lastCollision(): index of last collision experienced by this particle
  - UInt_t parentCollision(): index of parent collision creating this particle
    - Should <=lastCollision, because elastic scattering (A+B −> A+B) not recorded as parent collision
    - Not saved in UrQMD
    - To encode this and lastCollision, a variable "lstColl" in UrQMD is changed
  - UInt_t parentCollisionType()
    - = nElasticScatterings*100 + collisionType (>=0, <100) according to UrQMD
    - If only collision type required, calculate parentCollisionType%100
  - Int_t iTypeParent1(), Int_t i3Parent1()
  - Int_t iTypeParent2(), Int_t i3Parent2()
    - Parent i3 not saved in original UrQMD
    - To encode these four, a variable "origin" in UrQMD is changed

Introduction: class UrqmdTrack

- Useful functions:
  - UInt_t nCollisions()
  - UInt_t lastCollision(): index of last collision experienced
  - UInt_t parentCollision(): index of parent collision creatin
    - Should <=lastCollision, because elastic scattering (A+B −> A
    - Not saved in UrQMD
    - To encode this and lastCollision, a variable "lstColl" in UrQM
  - UInt_t parentCollisionType()
    - = nElasticScatterings*100 + collisionType (>=0, <100) accor
    - If only collision type required, calculate parentCollisionType
  - Int_t iTypeParent1(), Int_t i3Parent1()
  - Int_t iTypeParent2(), Int_t i3Parent2()
    - Parent i3 not saved in original UrQMD
    - To encode these four, a variable "origin" in UrQMD is change

Should be last collision number
(prevents reaction between 2 particles just reacted)

o13	o14	o15	o16	contents
		1		ind: index of particle (see CTOption(56))
1	1	2	1	t: time of particle
2	2	3	2 √	r_x: x coordinate
3	3	4	3 √	r_y: y coordinate
4	4	5	4 √	r_z: z coordinate
5	5	6	5	E: energy of particle
6	6	7	6 √	p_x: x momentum component
7	7	8	7 √	p_y: y momentum component
8	8	9	8 √	p_z: z momentum component
9	9	10	9 √	m: mass of particle
10	10	11	10 √	ityp: particle-ID
11	11	12	11 √	iso3: $2 \cdot I_3$ (see Section 1.2)
12	12	13	12 √	ch : charge of particle
13	13	14	13 √	parent collision number (see Table 10)
14	14	15	14 √	N_{coll} number of collisions
		16		S : strangeness
15	15		15 √	parent process type (see Table 11)
		17		history information (debugging only)
16				t^{fr} : freeze-out time of particle
17				r_x^{fr} : freeze-out x coordinate
18				r_y^{fr} : freeze-out y coordinate
19				r_z^{fr} : freeze-out z coordinate
20				E^{fr} : freeze-out energy of particle
21				p_x^{fr} : freeze-out momentum x component
22				p_y^{fr} : freeze-out momentum y component
23				p_z^{fr} : freeze-out momentum z component
	16*			τ_{dec} decay time of particle
	17*			τ_{form} formation time of particle
	18*			R_σ cross section reduction factor
	19*		√	unique particle number (not ID!)
			16* √	$ityp_1^{old}$: particle-ID of parent particle # 1
			17* √	$ityp_2^{old}$: particle-ID of parent particle # 2

Introduction: class UrqmdTrackEx

- Index always equal to UrqmdTrack

- In origin UrQMD, this information not saved for tracks belong to collisions
  - Can be saved via this code

- Useful functions:
  - Double_t tFreezeOut(),
  - Double_t xFreezeOut(), Double_t yFreezeOut(), Double_t zFreezeOut()
  - Double_t pxFreezeOut(), Double_t pyFreezeOut(), Double_t pzFreezeOut()
  - Double_t tauDecay(): decay time
  - Double_t tauForm(): formation time
  - Double_t rSigma(): cross section reduction factor

Introduction: class UrqmdTrackEx

- Index always equal to UrqmdTrack
- In origin UrQMD, this information not saved for track
  - Can be saved via this code
- Useful functions:
  - Double_t tFreezeOut(),
  - Double_t xFreezeOut(), Double_t yFreezeOut(), Double_
  - Double_t pxFreezeOut(), Double_t pyFreezeOut(), Doubl
  - Double_t tauDecay(): decay time
  - Double_t tauForm(): formation time
  - Double_t rSigma(): cross section reduction factor

o13	o14	o15	o16	contents
		1	√	ind: index of particle (see `CTOption(56)`)
1	1	2	1	t: time of particle
2	2	3	2 √	r_x: x coordinate
3	3	4	3 √	r_y: y coordinate
4	4	5	4 √	r_z: z coordinate
5	5	6	5	E: energy of particle
6	6	7	6 √	p_x: x momentum component
7	7	8	7 √	p_y: y momentum component
8	8	9	8 √	p_z: z momentum component
9	9	10	9 √	m: mass of particle
10	10	11	10 √	ityp: particle-ID
11	11	12	11 √	iso3: $2 \cdot I_3$ (see Section 1.2)
12	12	13	12 √	ch : charge of particle
13	13	14	13 √	parent collision number (see Table 10)
14	14	15	14 √	N_{coll} number of collisions
		16		S : strangeness
15	15		15 √	parent process type (see Table 11)
		17		history information (debugging only)
16			√	t^{fr} : freeze-out time of particle
17			√	r_x^{fr} : freeze-out x coordinate
18			√	r_y^{fr} : freeze-out y coordinate
19			√	r_z^{fr} : freeze-out z coordinate
20				E^{fr} : freeze-out energy of particle
21			√	p_x^{fr} : freeze-out momentum x component
22			√	p_y^{fr} : freeze-out momentum y component
23			√	p_z^{fr} : freeze-out momentum z component
	16*		√	τ_{dec} decay time of particle
	17*		√	τ_{form} formation time of particle
	18*		√	R_σ cross section reduction factor
	19*		√	unique particle number (not ID!)
			16* √	$\mathrm{ityp}_1^{\mathrm{old}}$: particle-ID of parent particle # 1
			17* √	$\mathrm{ityp}_2^{\mathrm{old}}$: particle-ID of parent particle # 2

Configuration File

- Before xxx
  - The same configuration as original UrQMD
  - f14 –> does not save f14 output file (#f14 –> saves f14 output file)

- After xxx
  - Specific configurationfor this code
  - #ske –> skips empty events (Ncoll + Ndecays == 0)
    - For non-empty events, Npart (only counts inelastic collisions) may be 0 (can be manually skipped when reading)
  - #qut –> quiet standard output
  - col –> does not save collision information
  - tke –> does not save track extended information

  - Removed since this version: esd –> does not use external seed for random

run > in.urqmd

```
1  pro 197 79
2  tar 197 79
3  imp -24
4  ecm 3
5
6  nev 10000
7  tim 50 50
8
9  #rsd 0
10 f13
11 f14
12 f15
13 f16
14 f19
15 f20
16
17 #cto 4 1
18 #cto 13 1
19 #cto 41 1
20 #cto 56 1
21 #cto 58 1
22
23 xxx
24
25 #ske
26 #qut
27 col
28 tke
```

How to save parent particle information

- 4-byte integer: "origin" (original definition)
  - = parentCollisionType + 100*(# elastic scatterings) + 1000*(|iType_parent1| + 1000*|iType_parent2|)
  - 0 <= parentCollisionType < 100, 0 <= |iType_ parent| < 1000
  - No "i3_parent" or sign of "iType_parent"

```
if ( (itypt(1).eq.ityp(inew(i))).and.
     (iso3t(1).eq.iso3(inew(i))) ) then
   origin(inew(i))=origint(1)+100
   uid(inew(i))=uidt(1)
elseif ( (itypt(2).eq.ityp(inew(i))).and.
         (iso3t(2).eq.iso3(inew(i))) ) then
   origin(inew(i))=origint(2)+100
   uid(inew(i))=uidt(2)
```

  - parentCollisionType = origin%100
  - # elastic scatterings = (origin/100)%10

ityp	nucleon	ityp	delta	ityp	lambda	ityp	sigma	ityp	xi	ityp	omega
1	N_{938}	17	Δ_{1232}	27	Λ_{1116}	40	Σ_{1192}	49	Ξ_{1317}	55	Ω_{1672}
2	N_{1440}	18	Δ_{1600}	28	Λ_{1405}	41	Σ_{1385}	50	Ξ_{1530}		
3	N_{1520}	19	Δ_{1620}	29	Λ_{1520}	42	Σ_{1660}	51	Ξ_{1690}		
4	N_{1535}	20	Δ_{1700}	30	Λ_{1600}	43	Σ_{1670}	52	Ξ_{1820}		
5	N_{1650}	21	Δ_{1900}	31	Λ_{1670}	44	Σ_{1775}	53	Ξ_{1950}		
6	N_{1675}	22	Δ_{1905}	32	Λ_{1690}	45	Σ_{1790}	54	Ξ_{2025}		
7	N_{1680}	23	Δ_{1910}	33	Λ_{1800}	46	Σ_{1915}				
8	N_{1700}	24	Δ_{1920}	34	Λ_{1810}	47	Σ_{1940}				
9	N_{1710}	25	Δ_{1930}	35	Λ_{1820}	48	Σ_{2030}				
10	N_{1720}	26	Δ_{1950}	36	Λ_{1830}						
11	N_{1900}			37	Λ_{1890}						
12	N_{1990}			38	Λ_{2100}						
13	N_{2080}			39	Λ_{2110}						
14	N_{2190}										
15	N_{2200}										
16	N_{2250}										

Table 1: Baryon-itypes used in **UrQMD**. Antibaryons carry a negative sign.

ityp	0^{-+}	ityp	1^{--}	ityp	0^{++}	ityp	1^{++}	ityp	charmed
101	π	104	ρ	111	a_0	114	a_1	133	D
106	K	108	K^*	110	K_0^*	113	K_1^*	134	D^*
102	η	103	ω	105	f_0	115	f_1	135	J/Ψ
107	η'	109	ϕ	112	f_0^*	116	f_1'	136	χ_c
ityp	1^{+-}	**ityp**	2^{++}	**ityp**	$(1^{--})^*$	**ityp**	$(1^{--})^{**}$	137	Ψ'
122	b_1	118	a_2	126	ρ_{1450}	130	ρ_{1700}	138	D_s
121	K_1	117	K_2^*	125	K_{1410}^*	129	K_{1680}^*	139	D_s^*
123	h_1	119	f_2	127	ω_{1420}	131	ω_{1662}		
124	h_1'	120	f_2'	128	ϕ_{1680}	132	ϕ_{1900}		

How to save parent particle information

- 4-byte integer: "origin" (new definition)
  - = parentCollisionType + 100*(# elastic scatterings)
    + 1000*((i3_parent1+3)+7*(t_iType_parent1+100))
    + 1400000*((i3_parent2+3)+7*(t_iType_parent2+100))
  - 0 <= parentCollisionType < 100, 0 <= i3_parent+3 <= 6
  - t_iType_parent=iType_parent; if(>=100) −=40; if(<=−100) +=40
    - 0 < t_iType_parent +100 < 200
  - origin < 1.96e9 (4-byte integer max ~ 2.1e9)

  - !!!Also affects *.txt output

  - parentCollisionType = origin%100
  - # elastic scatterings = (origin/100)%10

ityp	nucleon	ityp	delta	ityp	lambda	ityp	sigma	ityp	xi	ityp	omega
1	N_{938}	17	Δ_{1232}	27	Λ_{1116}	40	Σ_{1192}	49	Ξ_{1317}	55	Ω_{1672}
2	N_{1440}	18	Δ_{1600}	28	Λ_{1405}	41	Σ_{1385}	50	Ξ_{1530}		
3	N_{1520}	19	Δ_{1620}	29	Λ_{1520}	42	Σ_{1660}	51	Ξ_{1690}		
4	N_{1535}	20	Δ_{1700}	30	Λ_{1600}	43	Σ_{1670}	52	Ξ_{1820}		
5	N_{1650}	21	Δ_{1900}	31	Λ_{1670}	44	Σ_{1775}	53	Ξ_{1950}		
6	N_{1675}	22	Δ_{1905}	32	Λ_{1690}	45	Σ_{1790}	54	Ξ_{2025}		
7	N_{1680}	23	Δ_{1910}	33	Λ_{1800}	46	Σ_{1915}				
8	N_{1700}	24	Δ_{1920}	34	Λ_{1810}	47	Σ_{1940}				
9	N_{1710}	25	Δ_{1930}	35	Λ_{1820}	48	Σ_{2030}				
10	N_{1720}	26	Δ_{1950}	36	Λ_{1830}						
11	N_{1900}			37	Λ_{1890}						
12	N_{1990}			38	Λ_{2100}						
13	N_{2080}			39	Λ_{2110}						
14	N_{2190}										
15	N_{2200}										
16	N_{2250}										

Table 1: Baryon-itypes used in **UrQMD**. Antibaryons carry a negative sign.

ityp	0^{-+}	ityp	1^{--}	ityp	0^{++}	ityp	1^{++}	ityp	charmed
101	π	104	ρ	111	a_0	114	a_1	133	D
106	K	108	K^*	110	K_0^*	113	K_1^*	134	D^*
102	η	103	ω	105	f_0	115	f_1	135	J/Ψ
107	η'	109	ϕ	112	f_0^*	116	f_1'	136	χ_c
ityp	1^{+-}	**ityp**	2^{++}	**ityp**	$(1^{--})^*$	**ityp**	$(1^{--})^{**}$	137	Ψ'
122	b_1	118	a_2	126	ρ_{1450}	130	ρ_{1700}	138	D_s
121	K_1	117	K_2^*	125	K_{1410}^*	129	K_{1680}^*	139	D_s^*
123	h_1	119	f_2	127	ω_{1420}	131	ω_{1662}		
124	h_1'	120	f_2'	128	ϕ_{1680}	132	ϕ_{1900}		

How to store parent collision information

- Original UrQMD saves serial number of last collision
  - "lstColl" in code; parent collision number in users guide (better to say last collision number)
  - To prevent reaction between 2 particles from the same collision
  - If a particle goes through elastic collision (A+B −> A+B), "lstColl" is recorded, and its parent collision (in which collision the particle is produced) is not known

- In this code, both last collision and parent collision are saved
  - parentCollision <=lastCollision, because elastic scattering not recorded as parent collision
  - "lstColl" changed to encode both quantities (decoded when required in UrQMD simulation)
    - lastCollision = integer((sqrt(1+8.0d0*lstcoll)-1)/2)
    - For odd "lastCollision", parentCollision = lstColl-(lastCollision+1)/2*lastCollision
    - For even "lastCollision", parentCollision = lstColl-lastCollision/2*(lastCollision+1)

- !!!Also affects *.txt output

Check

```
root [46] u->collision(2634)->print()
5.0000e+01      9.2480e-01      0.0000e+00      0.0000e+00      1.0897e-02      20      1       2       3       10190
root [47] u->track(10190)->print()
5543    0       2634    104     0       0       -2      9.2490e-01      6.2051e+00      4.9854e+00      -4.7102e+01     3
.0487e-01       3.3380e-01      -2.9844e+00     1       2575    2575    20      -117    1       0       0
root [48] u->track(10191)->print()
5666    0       2634    101     2       1       0       1.3800e-01      6.2051e+00      4.9854e+00      -4.7102e+01     1
.4145e-01       4.8468e-01      -2.7690e+00     2       2634    2634    20      104     0       0       0
root [49] u->track(10192)->print()
5667    0       2634    101     -2      -1      0       1.3800e-01      6.2051e+00      4.9854e+00      -4.7102e+01     1
.6339e-01       -1.5085e-01     -2.1552e-01     1       2634    2634    20      104     0       0       0
```

```
      1       2  20    2634  50.000  0.9249E+00  0.0000E+00  0.0000E+00  0.1090E-01
2458  0.50000000E+02  0.62052729E+01  0.49856325E+01 -0.47099683E+02  0.31569740E+01  0.30484766E+00  0.33381796E+00 -0.29844185E+01  0.92485896E+00      104  0  0    2575    1  0         117020
2458  0.50000000E+02  0.62052729E+01  0.49856325E+01 -0.47099683E+02  0.28179373E+01  0.14146304E+00  0.48467783E+00 -0.27688991E+01  0.13800000E+00      101  2  1    2634    2  0         104020
2523  0.50000000E+02  0.62052729E+01  0.49856325E+01 -0.47099683E+02  0.33903666E+00  0.16338462E+00 -0.15085987E+00 -0.21551943E+00  0.13800000E+00      101 -2 -1    2634    1  0         104020
```

Check

```
root [51] u->collision(1759)->print()
2.1062e+01      2.7930e+00      8.8867e+00      2.1226e+00      2.7710e+00      26      2       2       4       7333
root [52] u->track(7333)->print()
4063    0       1759    101     2       1       0       1.3800e-01      -2.1704e+00     2.7202e+00      -2.0898e+01     -
2.0889e-01      -1.1183e-01     -7.8552e-01     1       1742    1742    15      17      -1      16      1
root [53] u->track(7334)->print()
3512    0       1759    17      1       1       -1      1.2320e+00      -2.2847e+00     2.9160e+00      -2.0926e+01     6
.0555e-02       2.7295e-01      -6.6250e+01     7       1366    1366    15      17      1       17      -1
root [54] u->track(7335)->print()
3512    0       1759    17      1       1       -3      1.2320e+00      -2.2847e+00     2.9160e+00      -2.0926e+01     7
.0203e-01       2.7917e-01      -5.7695e+01     8       1759    1366    115     17      1       17      -1
root [55] u->track(7336)->print()
4063    0       1759    101     2       1       0       1.3800e-01      -2.1704e+00     2.7202e+00      -2.0898e+01     -
8.5046e-01      -1.1807e-01     -9.3301e+00     2       1759    1742    115     17      -1      16      1
```

```
      2       2  26   1759  21.062  0.2793E+01  0.8887E+01  0.2123E+01  0.2771E+01
1869  0.21061567E+02 -0.21704844E+01  0.27203925E+01 -0.20897817E+02  0.83200094E+00 -0.20889727E+00 -0.11183377E+00 -0.78551684E+00  0.13800000E+00      101  2  1    1742  1  0       16017015
 351  0.21061567E+02 -0.22844273E+01  0.29161623E+01 -0.20926401E+02  0.66255078E+02  0.60558097E-01  0.27293928E+00 -0.66243033E+02  0.12320000E+01       17  1  1    1366  7  0       17017015
 351  0.21061567E+02 -0.22844273E+01  0.29161623E+01 -0.20926401E+02  0.57715714E+02  0.70206737E+00  0.27917483E+00 -0.57697617E+02  0.12320000E+01       17  1  1    1759  8  0       17017115
1869  0.21061567E+02 -0.21704844E+01  0.27203925E+01 -0.20897817E+02  0.93713649E+01 -0.85040655E+00 -0.11806933E+00 -0.93309327E+01  0.13800000E+00      101  2  1    1759  2  0       16017115

      2       4  15   1742  20.573  0.4078E+01  0.4000E+02  0.7113E+01  0.3180E+01
 289  0.20573349E+02 -0.23855281E+01  0.17656513E+01 -0.20424527E+02  0.17535527E+02 -0.78342021E+00  0.14621490E-01 -0.17475502E+02  0.12196703E+01      17 -1  0    1740  5  0       1101110
 231  0.20573349E+02 -0.20479036E+01  0.27860165E+01 -0.20436877E+02  0.47860026E+02  0.36748606E-01 -0.68083006E+00 -0.47785087E+02  0.25889575E+01      16  1  1    1462  7  0       1123227
 289  0.20573349E+02 -0.23855281E+01  0.17656513E+01 -0.20424527E+02  0.21800109E+02 -0.89090823E+00  0.53665146E-02 -0.21747027E+02  0.12320000E+01      17 -1  0    1742  6  0       16017015
 231  0.20573349E+02 -0.20479036E+01  0.27860165E+01 -0.20436877E+02  0.36313948E+02  0.52319997E+00 -0.39157334E+00 -0.36284463E+02  0.13090051E+01      17 -1  0    1742  8  0       16017015
1871  0.20573349E+02 -0.20479036E+01  0.27860165E+01 -0.20436877E+02  0.64494956E+01 -0.17006607E+00 -0.16816797E+00 -0.64435818E+01  0.13800000E+00      101  0  0    1742  1  0       16017015
1872  0.20573349E+02 -0.20479036E+01  0.27860165E+01 -0.20436877E+02  0.83200094E+00 -0.20889727E+00 -0.11183377E+00 -0.78551684E+00  0.13800000E+00      101  2  1    1742  1  0       16017015
```

Update on writting

- Add forward compatibility for ROOT version < 6.30

```
#if ROOT_VERSION_CODE >= ROOT_VERSION(6, 30, 00)
        outFile->SetBit(TFile::k630forwardCompatibility);
#endif
```

- Error when ROOT files written by ROOT >= 6.30 are read by ROOT < 6.30
  - Solution 1: file->SetBit(TFile::k630forwardCompatibility);
  - Solution 2: put TFile.v630forwardCompatibility: true in ~/.rootrc
    - Fully valid for ROOT 6.32 (>= 6.32.10) and ROOT 6.34 (>=6.34.04)
    - ROOT on ui is 6.32.02

Known Issues

- Negligible particles (such as some c-baryons) do not have UrQMD iType indices
  - In Original UrQMD, PDG ID saved in iType (4-byte integer)
  - In this code, only values between ± 100 are reserved for iType
    - iType == 0 is saved for these particles (can be identified using mass)

- Some UrQMD iType indices may not correspond to PDG ID
  - Current data structure (iType, i3) not affected

- Particles of the same species in UrQMD may not have the same masses
  - For proton, UrQMD mass = 0.938, some protons may < 0.938

Known Issues

- Different simulation results from CentOS 7 gfortran 4.8.5 and Ubuntu gfortran 11+
  - Unknow reason

- UrQMD simulation using some input seeds may be broken (exit with error)
  - Both original UrQMD and this code
  - Solution: input a new seed and resubmit the job

Introduction: $Location/UrqmdUtility

- Contents:
  - UrqmdUtility.{cxx,h}
  - GNUmakefile
    - Compiles Fortran $Location/urqmd/{blockres.f,error.f,ityp2pdg.f,upmerge.f} and C++ code in this directory to (lib)UrqmdUtility.so

- Useful functions:
  - static Int_t getPdgId(Int_t iType, Int_t i3)
  - static Int_t getPdgId(std::pair<Int_t, Int_t> id)
  - static TString getParticleName(Int_t iType)
  - static void getUrqmdId(Int_t& iType, Int_t& i3, Int_t pdgId)
  - static std::pair<Int_t, Int_t> getUrqmdId(Int_t pdgId)
  - static Int_t getUrqmdIType(Int_t pdgId)
  - static Int_t getUrqmdI3(Int_t pdgId)

Introduction: $Location/PdgData

- Contents:
  - ParticleDataEntry.{cxx,h}
  - PdgData.{cxx,h}
  - GNUmakefile
    - Compiles code in this directory to (lib)PdgData.so

- Useful functions:
  - PdgData: static const ParticleDataEntry* get(const Int_t pid)
  - ParticleDataEntry: Int_t pid(): PDG particle ID
  - ParticleDataEntry: Int_t spin(): double spin
  - ParticleDataEntry: Int_t charge(): triple electric charge
  - ParticleDataEntry: Int_t color()
  - ParticleDataEntry: Double_t mass()

Usage

- By ROOT: gSystem->Load($Location/UrqmdDst/libUrqmdDst.so");
- By GCC: -L$Location/UrqmdDst -lUrqmdDst -Wl,-rpath,$Location/UrqmdDst

- How to read tree
  - UrqmdDst* u = new UrqmdDst();
  - TChain* c = u->initInputChain("PATH_OF_FILE");//*.root or *.list
  - for(Long64_t iEntry=0; iEntry<c->GetEntries(); iEntry++){
  - c->GetEntry(iEntry);
  - for(UInt_t iTimestep=0; iTimestep<u->numberOfTimesteps(); iTimestep++){
  - const UrqmdTimestep* timestep = urqmdDst->timestep(iTimestep);
  - for(UInt_t iTrack=0; iTrack<timestep->nTracks(); iTrack++){
  - const UrqmdTrack* track = u->track(timestep->startTrack()+iTrack);
  - …

Usage

- By ROOT: gSystem->Load($Location/UrqmdUtility/libUrqmdUtility.so");
- By GCC: -L$Location/UrqmdUtility -lUrqmdUtility -Wl,-rpath,$Location/UrqmdUtility

- By ROOT: gSystem->Load($Location/PdgData/libPdgData.so");
- By GCC: -L$Location/PdgData -lPdgData -Wl,-rpath,$Location/PdgData

```
root [2] PdgData::get(UrqmdUtility::getPdgId(101,2))->mass()
(double) 0.13957000
root [3] UrqmdUtility::getUrqmdId(UrqmdUtility::getPdgId(101,2))
(std::pair<Int_t, Int_t>) { 101, 2 }
root [4] UrqmdUtility::getParticleName(UrqmdUtility::getUrqmdIType(2212))
(TString) "Nukleon"[7]
root [5] UrqmdUtility::getParticleName(UrqmdUtility::getUrqmdIType(-2212))
(TString) "*Nukleon"[8]
```

Summary

- A few updates on UrQMD production and reading
- Almost all information can be accessed with as small disk space required as possible
- New version compatible with older ones
- New codes UrqmdUtility and PdgData help with reading UrQMD data
- Existing UrQMD samples in ui:/ustcfs/HICUser/fsi/MODEL/UrQMD/
  - Naming conventions: ProjectileTarget_energy(_configuration)
  - E.g., UU_2p1, AuAu_3_eos1